VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Systém technické analýzy burzovních dat

System for Technical Analysis

2014

Bc. Václav Němec

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Václav Němec**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Systém technické analýzy burzovních dat
System for Technical Analysis

Zásady pro vypracování:

Technická analýza se používá na předpovídání budoucích cenových pohybů na základě systematického zkoumání, analyzování a vyhodnocování minulých a současných dat. Je používána u všech finančních produktů, včetně cenných papírů, futures a úrokových produktů.
Cílem diplomové práce je vytvoření systému pro simulování obchodů na historických datech a optimalizace strategií při obchodování použitím biologicky inspirovaných algoritmů. Dále pak vytvoření obchodního indexu reprezentujícího "náladu trhu" za použití dat ze sociálních/zpravodajských sítí.

1. Rešerše automatických systémů obchodujících na základě technické analýzy.
2. Simulace obchodování a optimalizace strategií pomocí biologických algoritmů.
4. Simulace obchodování za použití navrženého indexu "nálady trhu".
5. Vyhodnocení, srovnání úspěšnosti nalezených strategií.

Seznam doporučené odborné literatury:

[1] Kendall K., Electronic and Algorithmic Trading Technology: The Complete Guide, Academic Press, 2007, ISBN 0123724910
[2] Vialar, T., Complex and Chaotic Nonlinear Dynamics: Advances in Economics and Finance, Mathematics and Statistics, Springer, 2009, ISBN 3540859772
[3] Johnson, B., Algorithmic Trading and DMA: An introduction to direct access trading strategies, 4Myeloma Press, 2010, ISBN 0956399207

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Mgr. Ing. Michal Krumnikl**

Datum zadání: 01.09.2013

Datum odevzdání: 07.05.2014

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě, 20. května 2014 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě, 20. května 2014 .....

Děkuji za podporu vedoucímu Mgr. Ing. Michalovi Krumniklovi. Zvláštní poděkování patří mé rodině a Hance za motivaci a podporu, kterou jsem od nich potřeboval.

Abstrakt

Cílem této diplomové práce je vytvoření simulačního systému pro simulaci automatického obchodování na burze, který umožňí vytvoření obchodní strategie pomocí technické analýzy, technických indikátorů a neuronové sítě a který umožní optimalizaci takto vytvořené obchodní strategie pomocí evolučně inspirovaných algoritmů, jako je genetický algoritmus nebo samo organizující se migrační algoritmus. Dále si práce klade za cíl navržení nového obchodního indexu reprezentujícího "náladu trhu" využívajícího texty zpravodajských zdrojů a výsledné porovnání vytvořených strategií.

Klíčová slova: Technická analýza, indikátory, SMA, RSI, automatické obchodní systémy, algoritmické obchodování, optimalizace, biologicky inspirované algoritmy, evoluční algoritmy, neuronové sítě, zpracování přirozené řeči, sentimentální analýza, nálada trhu, simulace, java

Abstract

The aim of this diploma thesis is creation of a system for simulation of automatic stock exchange. This system will enable planning of business strategy through technical analysis, technical indicators and neural network and will also provide optimalization of the business strategy through evolutionary algorithms such as genetic algorithm or self-organizing migration algorithm. The thesis then further focuses on designing new business index representing "market mood" through text messages from news bulletin sources and eventual comparison of created strategies.

Keywords: Technical analysis, indicators, SMA, RSI, automatic trade system, algorithmic trading, optimization, evolutionary algorithm, neural network, natural language processing, semantic orientation, market mood, simulation, java

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Obchodování na burze je možnost, jak zúročit kapitál. Tato práce se zabývá přípravou obchodní strategie a některými způsoby, jak toho docílit simulací, pomocí které můžeme vytvořit robustní strategii s použitím historických burzovních dat a **optimalizačních metod**, popřípadě pomocí modelu **neuronové sítě**, která je dobrá pro predikci budoucích burzovních pohybů. Dále si práce klade za cíl navržení indexu **nálady trhu** založeném na analýze internetových zpráv o akciovém trhu, což je inovativní přístup, který by mohl být při obchodování užitečný.

V kapitole 2 je přiblížena problematika burzovního obchodování, respektive **technické analýzy** a jejího použití. Jsou zde uvedeny základní indikátory, které se využívají spolu se strategiemi a možnostmi jejich využití v technické analýze.

V současnosti existuje spousta nástrojů podporujících **automatické obchodování**, které zároveň umožňují obchodování pomocí technické analýzy, například robustní open source řešení *Algo Trader* a *Cyan Spring ATS*, která jsou zaměřena jak na malé obchodníky, tak na velké investiční společnosti. Rešerže řešení pro automatické obchodování je v kapitole 3

Optimalizace nabízí možnosti, jak **maximalizovat** výnosnost obchodní strategie. Téma optimalizace je popsáno v kapitole 4. Jsou zde zmíněny základní pojmy a principy optimalizace a je zde také popsán **genetický algoritmus** a **samo organizující se migrační algoritmus**, který je použit v praktické části této práce pro optimalizaci implementovaných obchodních strategií.

Proces rozhodování, zda-li nakoupit nebo prodat finanční instrument není jednoduchá úloha, je ovšem možné použít metody strojového učení a tuto úlohu *naučit* počítač. K tomuto účelu se dá využít modelu **neuronové sítě**, který jako jeden ze svých běžných případů užití, slouží pro predikci časových řad. Neuronovou síť je možné použít pro predikci následující ceny z cen předchozích, popřípadě z hodnot indikátorů. Principy neuronových sítí jsou popsány v kapitole 5.

Jedna z možností, kterou lze použít při analyzování situace na trhu, je analýza dat ze sociálních a zpravodajských sítí. Takto získaná data mohou být podrobena **zpracování přirozené řeči** počítačem a může s nimi být počítáno při rozhodování o provedení obchodní transakce. Principy přirozeného zpracování řeči a jeho využití v sentimentální analýze jsou popsány v kapitole 6.

Existují různé prostředky pro přípravu dobré obchodní strategie, ale jako každý hotový software mají předurčené limity. Implementování řešení pro práci s obchodních strategiemi může být výhodné v několika směrech, například umožní porovnání obchodních algoritmů na stejném simulačním modelu. Strategie vytvořené a optimalizované takovým systémem mohou být mnohem komplexnější a nejsou vázány žádnými architektonickými omezeními. Kapitola 7 popisuje vytvoření systému pro technickou analýzu, který využívá technických indikátorů, neuronovou síť, zpracování přirozené řeči (textu) a umožňuje optimalizaci. V této kapitole je celý simulační systém popsán.

Navazující kapitola 8 se zabývá vyhodnocením výsledků tří vytvořených strategií používající indikátory, neuronovou síť a vytvořený index *nálady trhu*. Je provedena *korelační analýza* pro hodnotu indexu a příslušné ceny akcie a je provedeno vyhodnocení výsledků.

2 Technická analýza

Technická analýza je metoda, na které se dá založit automatická obchodní strategie[1]. Technická analýza se zabývá studiem vlastností finančních produktů, jako je cena, volatilita[2], objem (volume)[3] atp. Hlavním nástrojem technické analýzy je práce s grafy. Podle [4] se vznik moderní technické analýzy datuje okolo roku 1900. Charles Dow je autorem teorie (Dowova teorie), ve které jsou popsány základní principy technické analýzy. Dow například popsal přirozený vývoj ceny v trendech, vliv objemu obchodů na změnu ceny, úrovně support a resistance[4] a další.

Na rozdíl od fundamentální analýzy, která se zabývá vlivem ekonomiky na trh, je technická analýza založena jenom na zkoumání informací z grafů. Při zkoumání grafů můžeme například hledat různé cenové formace, popřípadě můžeme statisticky zpracovávat cenu či jiné informace. Statistika hraje v technické analýze velkou roli ve formě technických indikátorů. Indikátory jsou hojně využívány pro predikci vývoje ceny.

2.1 Indikátory

Podle [10] je technický indikátor série datových bodů, které se vypočítají z ceny aplikováním vzorce. Kromě ceny můžeme hodnotu indikátoru počítat například z objemu transakcí. Indikátory slouží k lepšímu znázornění toho, co se na trhu právě odehrává. Nabízí další perspektivu, která není přímo čitelná jenom z údajů o ceně. Indikátory obecně slouží k item upozornění, potvrzení a predikování. Indikátory se dělí na:

- trendové indikátory,
- vedoucí indikátory.

2.2 Trendové indikátory (Lagging Indicators)

Podle [10] následují tyto indikátory hlavní trend. Indikátory tohoto typu pracují nejlépe, pokud trh silně trenduje. Mezi základní trendové indikátory patří klouzavé průměry SMA (Simple Moving Average), EMA (Exponential Moving Average), MACD (Moving Average Convergence Divergence).

2.2.1 SMA - Simple Moving Average

Toto je nejjednodušší indikátor. Podle [10] je SMA indikátor, který počítá průměr posledních n hodnot. Pokud je volatilita vysoká, umožní nám SMA *vyhladit* data, a tím nabízí možnost jednodušeji identifikovat hlavní trend. Základní strategie založená na tomto indikátoru je následující: Pokud se cena protne s hodnotou indikátoru směrem dolů, znamená to, že je vhodné nakoupit. Pokud se cena protne s hodnotou indikátoru směrem nahoru, znamená to, že je vhodné prodat.

[1] Strategie, která provádí rozhodování o nákupu či prodeji automaticky bez zásahu člověka, popřípadě s jeho dohledem

[2] Jak je uvedeno v [1], je volatilita statistická míra rozptylu. Čím vyšší hodnota, tím více sledovaná hodnota v čase kolísá, čím menší hodnota, tím je kolísání menší.

[3] Počet uzavřených obchodů v čase

[4] Support nebo resistance je úroveň ceny pod, respektive nad, za kterou nechce žádný účastník trhu prodávat, respektive nakupovat, uvádí [2]

Obrázek 1: Znázornění indikátoru SMA[5]

Tato jednoduchá strategie může generovat spoustu falešných signálů k prodeji a nákupu. Indikátor je zobrazen na obrázku 1.

2.3 Vedoucí indikátory (Leading Indicators)

Podle [10] jsou tyto indikátory navrženy tak, aby vedly (predikovaly) cenové pohyby. Většina zachycuje momentum ceny za poslední periodu. Mezi tyto indikátory patří například indikátor RSI (Relative Strength Index), Momentum, Stochastic Oscillator.

2.3.1 RSI - Relative Strength Index

Podle [6, 10] nabývá indikátor hodnot od nuly do sta. RSI porovnává průměrnou změnu ceny kladných časových period s průměrnou změnou ceny záporných period. [6] uvádí, že je nutné si stanovit dvě pracovní hladiny. První hladina značí překoupený trh[6] a tradičně se volí jako pásmo, ve kterém hodnoty nabývají sedmdesáti a výš. Druhá hladina značí přeprodaný trh[7] a jde o pásmo, ve kterém hodnoty nabývají třiceti a níže. V praxi se setkáme s různými způsoby využití tohoto

[5] Převzato z `http://www.financnik.cz/komodity/manual/komodity-klouzave-prumery.html`

[6] Nikdo už nechce nakupovat

[7] Nikdo už nechce prodávat

indikátoru, ale základní strategie založená na tomto indikátoru generuje signál k nákupu v případě, kdy je trh *přeprodaný* a signál k prodeji je generován v případě, kdy je trh *překoupený*.

$$RS = \frac{G(n)}{L(n)} \tag{1}$$

$$RSI = 100 - \frac{100}{1 + RS} \tag{2}$$

Výpočet hodnoty *relativní síly* se provádí vzorcem 1, kde $G(n)$ je průměrná hodnota zisků za periodu n a $L(n)$ je průměrná hodnota ztrát za periodu n, hodnota indikátoru se pak počítá vzorcem 2.

3 Automatické obchodování

Jedním z cílů automatického obchodování (nebo také automatických obchodních systémů) je odstranit lidský faktor z procesu rozhodování o nákupu či prodeji. Podle [11] je automatický obchodní systém: "jakákoli strategie, která se vykonává nezávisle na vašem rozhodnutí, nicméně podle vámi zadaných pravidel. Není proto nutné sledovat celý den trh a čekat na signál, AOS to udělá za vás. AOS je ve své podstatě shluk pravidel, která definují vstupy a výstupy z trhu a program poté sám zadává příkazy na trh. Systém může být postaven prakticky na čemkoliv, co vám funguje. Například systém, který vstupuje a vystupuje na základě křížení klouzavých průměrů, svíčkových formací, arbitráže, korelace apod." V této kapitole jsou zmíněna existující řešení pro tuto oblast.

3.1 Algo Trader

Algo Trader je platforma, která umožňuje automatické obchodování na trhu FOREX[8], opcí[9], akcií[10], komodit[11] a dalších finančních produktů. Je dostupná ve dvou edicích:

- Open Source edice - tato verze je spravována komunitou a neobsahuje všechny funkce. Slouží jako základ pro placenou edici.
- Enterprise edice - tato verze obsahuje vše, co Open Source edice a také mnoho dalších funkcí viz Product Features

Nástroj umožňuje automatické obchodování na základě kvantitativních obchodních strategií, což znamená, že obchodní strategie, kterou lze formulovat počítačovým programem, jde automatizovat. Algo Trader podporuje velmi rychlé zpracování a analýzu velkých objemů obchodních dat. Základem jsou frameworky: Esper pro zpracování událostí, AndroMDA pro modelem řízenou architekturu, Hibernate pro perzistentní vrstvu, Spring Framework pro docílení servisně orientované architektury. Jak je znázorněno na obrázku 2, je architektura systému Algo Trader sestavena ze dvou úrovní. Obchodní vrstva (Trading Framework) slouží jako základ pro běh strategií. Vrstva je zodpovědná za:

- perzistování dat do databáze
- připojení na brokera
  - načítání živých obchodních dat
  - zadávání obchodních příkazů
- risk management

[8] Mezinárodní měnový trh

[9] Jedná se o obchodování předkupních práv, nikoli produktu, jak uvádí [3]

[10] Cenných papírů dané společnosti

[11] Kukuřice, zlato, káva, atp.

[12] Převzato z dokumentace Algo Traderu `https://code.google.com/p/algo-trader/wiki/AlgoTraderDocumentation`

Obrázek 2: Architektura systému Algo Trader[12]

- portfolio management
- performance measurment

Na předchozí vrstvě může být nainstalováno jakékoliv množství strategií. Strategie může využívat technické indikátory a specifická obchodní pravidla pro vyhodnocení obchodních signálů.

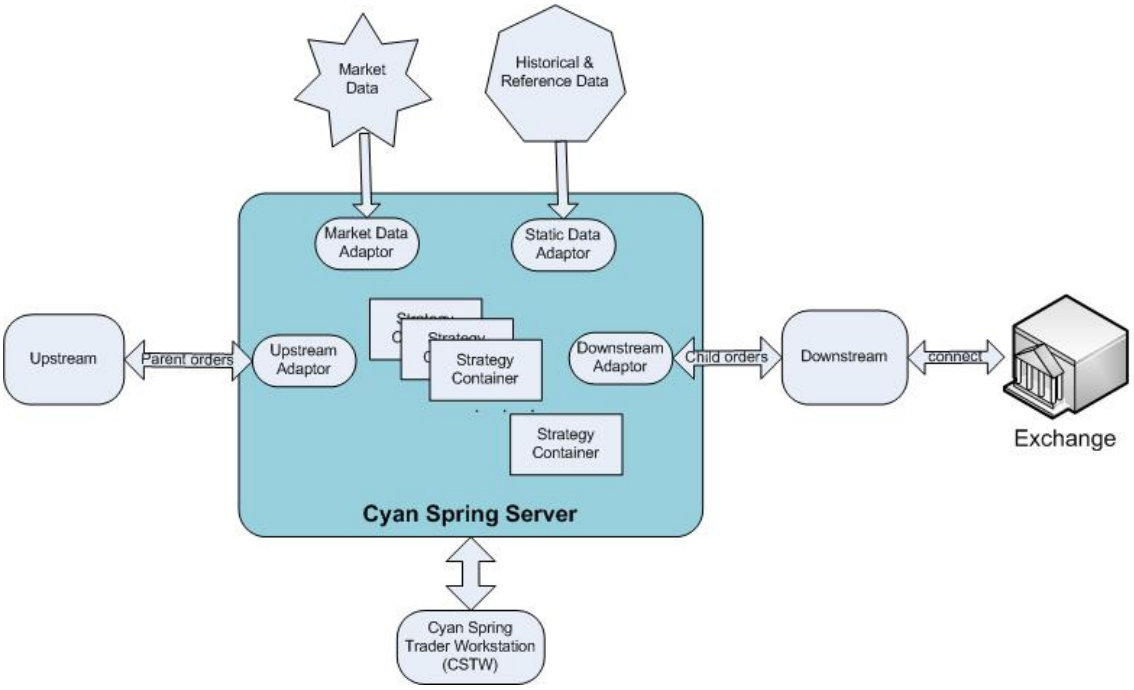
3.2 Cyan Spring ATS

Informace o Cyan Spring ATS byly převzaty z [12]. Cyan Spring ATS je open source algoritmický obchodní systém (algorithmic trading platform). Jeho cílem je poskytnout řešení pro automatické obchodování pro investice bank, správce fondů a individuální tradery. Cyan Spring kombinuje algoritmické obchodování spolu s order managementem do jednoho integrovaného řešení, které umožňuje rapidní vývoj obchodních strategií. Jak je znázorněno na obrázku 3, je architektura systému rozdělena do několika komponent:

- Cyan Spring Trader Workstation - jedná se o klientskou aplikaci pro připojení a monitorování systému,
- Market Data Adaptor - umožňuje přístup k živým datům,
- Static Data Adaptor - umožňuje přístup k historickým a referenčním datům,
- Downstream, Upstream a Exchange - umožňují zadávání příkazů pro brokera,
- Strategy Container - obsahuje implementaci strategie.

Cyan Spring ATS může fungovat v distribuovaném prostředí, ve kterém si více instancí systému rozděluje zátěž. Celé řešení je založeno na následujících technologiích: Active MQ - pro výměnu asynchronních zpráv, QuickFIX/J - protokol pro výměnu finančních zpráv, Spring Framework - pro integrační vrstvu, Rich Client Platform - pro uživatelské rozhraní, XStream - pro serializování dat, Simple Logging Facade for Java - abstrakce pro logování, Apache log4j - knihovna pro logování, Apache Derby - databázový stroj.

[13]Převzato z dokumentace Cyan Spring ATS `http://www.cyanspring.com/Architecture.php`



Obrázek 3: Architektura systému Cyan Spring ATS[13]

3.3 Arizona Financial Text System (AZFinText)

Podle [17] je AZFinText projektem zabývajícím se předpovědí budoucí ceny akcií na základě finančních článků. Analyzováním důležitých zpráv a zaměřením se na konkrétní slovní druhy, portfolio a váhy termínů je systém odlišný od ostatních, které se zabývají kvantitativním přístupem. Autor také uvádí, že systém předčil většinu nejlepších systémů.

4 Optimalizace

Podle [8] jsou optimalizační algoritmy mocným nástrojem pro řešení mnoha problémů inženýrské praxe. Využívají se tam, kde je řešení analytickou cestou nemožné či nevhodné. Optimalizace spočívá v nalezení takové kombinace argumentů optimalizovaných funkcí, které je optimální pro řešení problému. V burzovním obchodování se optimalizace využívá pro nalezení takových parametrů obchodních strategií, které maximalizují zisk a minimalizují riziko. Argumenty optimalizovaných funkcí mohou nabývat různých typů:

- celočíselné,
- reálné,
- komplexní

4.1 Účelová funkce

Správné definování účelové funkce je kritické při hledání optimálního řešení. Jak uvádí [8], optimalizační funkcí rozumíme funkci, u které se snažíme nalézt maxima či minima a příslušné argumenty těchto hodnot. Na každou účelovou funkci lze nahlížet jako na geometrický problém, kde je počet dimenzí plochy řešení dán počtem optimalizovaných argumentů, celkový počet rozměrů prohledávaného prostoru je pak o jedna vyšší, poslední prostor je návratová hodnota účelové funkce.

4.2 Evoluční algoritmy

V posledních desetiletích byla vyvinuta třída algoritmů se specifickým názvem "evoluční algoritmy". Evoluční algoritmy jsou výkonné a dokáží řešit komplexní problémy, neexistuje však univerzální algoritmus, který by dokázal vyřešit každý problém. Různé algoritmy se hodí pro různé problémy. Základní dělení evolučních algoritmů je následující:

- deterministické, tzn. algoritmy, u kterých je předem možné odvodit všechny kroky,
- stochastické, tzn. algoritmy, u kterých se využívá náhodných operací a tudíž nejsme schopni předem odvodit všechny kroky,
- komplexní.

4.2.1 Populace

V kontextu evolučních algoritmů je populace struktura jedinců stejného typu. Jedincem myslíme vektor hodnot argumentů účelové funkce. Jedince jsme schopni uspořádat na základě hodnoty účelové funkce. Evoluční algoritmy obecně pracují v krocích, ve kterých vytvářejí nové a nové populace.

4.2.2 Specimen

Specimen je vzor, popisuje typ a hranice hodnot jedinců. Je to kolekce elementů, udávajících limity možného řešení, které v evolučních algoritmech slouží pro tvorbu nových potomků. Každá hodnota, ze které se skládá specimen má tři vlastnosti:

- typ, zda se jedná o celočíselnou, reálnou, nebo komplexní hodnotu,
- minimum, nejnižší možná hodnota,
- maximum, nejvyšší možná hodnota.

4.2.3 Genetický algoritmus

Genetický algoritmus je nejznámější evoluční algoritmus. Vychází z teorie Charlese Darwina. Klasické genetické algoritmy pracují s dvojkovou reprezentací genetického kódu jedinců, můžeme se setkat i s jinou reprezentací, například reálnou. Genetický algoritmus pracuje v evolučních krocích, kterým říkáme generace. Pro každou generaci se musí vybrat množina rodičů, pseudonáhodně vybraných jedinců s nejlepší vhodností. Existuje několik algoritmů pro výběr rodičů na základě hodnoty fitness. Každá metoda se liší v určení pravděpodobnosti, podle které následně vybírá rodiče. Jakmile je hodnota pravděpodobnosti vypočítána, vybíráme jedince podle pseudonáhodného generování čísel. Detailně je postup popsán v [8].

4.2.3.1 Výběr pomocí rulety Jedná se o metodu pro výběr rodičů. Pravděpodobnost, že bude jedinec vybrán do množiny rodičů je rovna normalizované absolutní hodnotě jeho vhodnosti. Uveďme příklad: Máme tři jedince s vhodností 2, 8 a 10. Pravděpodobnosti výběrů jedinců jsou v pořadí 0.1, 0.4 a 0.5. Transformace vhodnosti na hodnotu pravděpodobnosti je detailně popsána v [8].

4.2.3.2 Rank Selection Jedná se o metodu pro výběr rodičů. Pří výběru pomocí této metody seřadíme jedince podle hodnoty fitness od nejhoršího po nejlepšího. Takto seřazeným jedincům přiřadíme čísla od jedné inkrementálně, jedná se v podstatě jednoduše o index, a pro získání pravděpodobnosti, že bude jedinec vybrán do množiny rodičů, tuto hodnotu podělíme aritmetickým součtem od jedné do počtu všech jedinců, ze kterých rodiče vybíráme.

4.2.3.3 Elitismus Elitismus v genetickém algoritmu slouží k uchování elitních jedinců s generace na generaci. Nejlepší jedinec, popřípadě jedinci, jsou automaticky vybráni, aby byli součástí další generace. Elitismus urychluje genetický algoritmus. V případě, že bude počet elitních jedinců přenášených z generace do generace příliš velký, respektive poměr k celkovému počtu jedinců překročí určitou mez, může genetický algoritmus vést k lokálním řešením.

4.2.3.4 Křížení Poté, co vybereme množinu rodičů, můsíme z této množiny vytvořit nové potomky. Jedním ze způsobů reprodukce je křížení. Vstupem křížení jsou dva, popřípadě více, jedinci a výsledný jedinec je vytvořen kombinací genetického kódu rodičů. Křížení existuje jednobodové a vícebodové. V případě, že jsou jedinci reprezentováni dvojkovým kódem, jsou kombinovány celé

bloky nul a jedniček. V případě, že je jedinec reprezentován reálnými čísly, můžeme se na problém dívat podobně, a to tak, že budeme také zaměňovat jednotlivá reálná čísla popřípadě celé sekvence. Parametrem křížení je pravděpodobnost, která určuje, jak se budou jedinci křížit. Pravděpodobnost 0,5 říká, že je 50% šance, že se genetická informace vymění. V reprezentaci jedinců reálními čísly je možné provádět také aritmetické křížení, například na základě průměru, to znamená, že pokud nastane křížení, je genetická informace rodičů zprůměrována a předána potomkovi.

4.2.3.5 Mutace Mutace je nástrojem, který zavádí další nedeterminsmus do genetického algoritmu. Pro běh genetického algoritmu musíme zvolit pravděpodobnost, že nastane mutace. Pro dvojkovou reprezentaci se mutace provádí inverzí bitů na základě pravděpodobnosti. U reprezentace reálnými čísly to může být generování nového jedince podle normálního rozložení se středem v původní nezmutované hodnotě.

4.2.4 Samo organizující se migrační algoritmus

SOMA je algoritmus založený na práci s vektory. Oproti genetickému algoritmu, který pracuje s populacemi, pracuje SOMA s migračními koly. Lze jej řadit mezi evoluční algoritmy i přesto, že jeho myšlenka není založena na vytváření nových potomků, ale na jejich migraci. Lze jej taky zařadit mezi algoritmy hejnové (swarm intelligence). Informace o algoritmu SOMA byly čerpány z [8].

Princip algoritmu je založen na napodobení chování skupiny inteligentních jedinců, kteří kooperují při řešení společného problému. Samotný algoritmus existuje v několika variantách a byl postupně měněn. Robustnost algoritmu je srovnatelná s robustností jiných evolučních algoritmů.

V algoritmu neprobíhá tvorba nových řešení pomocí křížení jedinců, jak je tomu u jiných algoritmů, ale noví jedinci vznikají společnou spoluprací a migrací v prohledávaném prostoru možných řešení, proto se evolučnímu cyklu říká *migrační kolo* a ne *generace*.

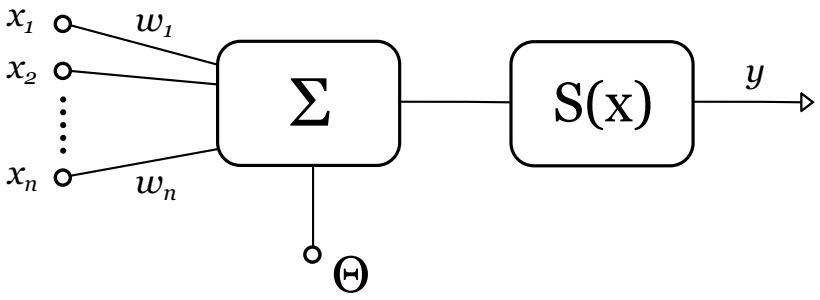
4.2.4.1 Inicializace Populace se na začátku běhu algoritmu vygeneruje pseudo-náhodně podle Specimenu, znamená to tedy, že vygenerovaná řešení jsou náhodně rozeseta v prostoru možných řešení.

4.2.4.2 Migrační kolo Každý jedinec je ohodnocen účelovou funkcí a nejlepší jedinec je zvolen za *leadera*. Nyní nastává proces podobný křížení u genetického algoritmu. Jedinci se začnou v prostoru řešení skokově přibližovat směrem k *leadrovi* o velikost definovanou parametrem **step** s tím, že si pamatují svou nejlepší pozici vzhledem k účelové funkci a pohybují se tak dlouho, dokud nedosáhnou pozice vypočtené z parametru *pathLength*. Pokud je hodnota parametru pathLength 1, znamená to, že jedinec doskáče maximálně na pozici leadera. V případě, že je hodnota parametru pathLength 2, znamená to, že jedinec doskáče 2x dále (až za leadera), než v předchozím případě. Z definice parametru pathLength vyplývá, že krok by měl být nastaven tak, aby se jedinec nesetkal s leaderem, například na hodnotu 0.11. Předchozí text popsal základní myšlenku, chybí zde ovšem obdoba mutace. Původně je algoritmus definován tak, že se během skoku jedinec posouvá o hodnotu definovanou parametrem step na každé ose. Zavedeme další parametr, tím bude parametr **PRT** (jako perturbace, rušení), který nabývá hodnot od 0 do 1. Při každém skoku musíme pro každou

osu (dimenzi) vygenerovat pseudonáhodné desetinné číslo on nuly do jedné a pokud je hodnota vygenerovaného čísla větší než hodnota parametru PRT, skok se pro danou dimenzi provede, pokud je menší, skok se pro danou dimenzi neprovede. Výsledkem může být, že se jedinec přiblíží ve dvou dimenzích a ve třetí nikoliv atp. Zmínili jsme parametry pathLength, step a PRT. Kromě těchto parametrů existují ještě parametry **D**, což je počet dimenzí daného problému, **popSize**, což je velikost populace, **Migrace**, což je ukončovací parametr značící maximální počet migračních kol a parametr **MinDiv**, což je volitelný ukončovací parametr, který definuje jak velký je povolený rozdíl mezi nejlepším a nejhorším jedincem populace. V případě, že je rozdíl menší, než hodnota parametru, je výpočet ukončen.

5 Neuronové sítě

Přesněji řečeno *umělé neuronové sítě* jsou výpočetním modelem, který napodobuje *biologické neuronové sítě*. Neuronové sítě se využívají pro predikci časových řad a je možné je využít pro predikování budoucího vývoje cen na burze. Jak uvádí [13], jsou neuronové sítě inspirovány biologickými neuronovými sítěmi, a tím jsou předurčeny k principiálně stejnému či podobnému chování. Nelze vytvořit umělý lidský mozek, ale lze simulovat některé funkce lidského myšlení. Neuronové sítě využívají distribuované paralelní zpracování informace při provádění výpočtů. Paměť neuronové sítě je spíše globální, než lokální. Znalosti jsou ukládány pomocí síly vazeb mezi neurony tak, že znalosti, vedoucí ke správné odpovědi, jsou posilovány a znalosti, vedoucí ke špatné odpovědi, jsou oslabovány. Stejně jako se učí lidský mozek, tak se učí i neuronové sítě. Ke své činnosti potřebují neuronové sítě tzn. trénovací množinu. Učení sítě pak rozdělujeme na učení s učitelem a bez učitele. Při učení s učitelem zná neuronová síť dopředu požadovaný výstup pro testovací vstup. U učení bez učitele tomu tak není a výstup pro trénovací vstup určen není. Základem pro pochopení neuronových sítí je model perceptronu a aplikace různých aktivačních funkcí pro základní problémy, jako jsou například kategorizace do dvou lineárně separovatelných množin a lineární regrese. Skutečné využití neuronových sítí pak přináší vícevrstvé a rekurentních neuronové sítě. Na obrázku 4 lze vidět model neuronu. Hodnoty x_1 až x_n jsou vstupní signály, w_1 až w_n jsou váhy propojení vstupního signálu do neuronu, Θ je práh neuronu. Vstupní hodnoty se nejdříve vynásobí příslušnými vahami a spolu s hodnotou *prahu* jsou sečteny a výsledný signál je poslán na vstup aktivační funkce neuronu, což je skutečný výstup neuronu Y.

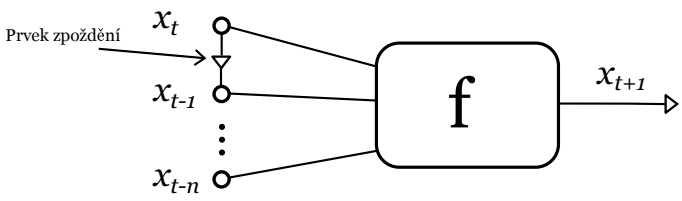


Obrázek 4: Model neuronu

5.1 Predikce časových řad

Predikování budoucích hodnot časové řady je problém, ve kterém můžeme využít vícevrstvé[14] neuronové síťě. Informace o predikci časových řad byly čerpány z [14]. Časovou řadu si v našem případě můžeme představit jako sekvenci čísel. Pro jiné účely může být časová řada definována jako sekvence vektorů, my si ale vystačíme s čísly. Příkladem časové řady může být hodnota akcií, sluneční aktivita, teplota na určitém místě, počet nakažených lidí žloutenkou. Časové řady monou být diskrétní nebo spojité. Příkladem diskrétní časové řady je uzavírací hodnota akcií pro hodinové

[14] Jedná se o sítě, které jsou složeny z několika dopředně propojených vrstev neuronů, kde je výstup každého neuronu spojen se všemi vstupy neuronů v následující vrstvě



Obrázek 5: Blokové schéma pro predikci časové řady

intervaly. Spojitý signál musíme převést na diskrétní signál *vzorkováním* o dané frekvenci. Studiem časových řad se zabývají tři disciplíny:

- informační teorie (Information Theory) - zabývá se stochastickými časovými řadami,
- teorie dynamických systémů (Dynamical Systems Theory) - zabývá se nelineárními deterministickými řadami,
- zpracování digitálních signálů (Digital Signal Processing) - zabývá se lineárními časovými řadami jak deterministickými, tak stochastickými.

Podle typu využití rozdělujeme:

- predikci - zajímají nás budoucí hodnoty proměnné v čase,
- klasifikaci - zajímají nás třídy, například:
  - cena stoupne,
  - cena klesne,
  - cena se nezmění.
- transformaci - transformujeme jednu časovou řadu na druhou. Například cenu ropy na úrokovou sazbu.

Při predikování hodnoty vycházíme z předchozích hodnot, jak je znázorněno na obrázku 5. Hodnoty x_t až x_{t-n} jsou vstupy pro predikci, které šly v čase za sebou. f je systém, který provádí predikci a x_{t+1} je požadovaná predikovaná hodnota. Místo bloku f se při predikci dá využít neuronová síť.

6 Zpracování přirozené řeči

Podle definice z [15] je zpracování přirozeného jazyka disciplína zabývající se výpočetními metodami pro analyzování a reprezentování přirozeně se vyskytujícího se textu v jedné či více úrovních lingvistické analýzy tak, jak jej zpracovávají lidé přirozeně, pro účely specifických aplikací. Existují různé metody či techniky pro analyzování přirozeného textu, které jsou vhodné pro různé druhy analýzy. Pojmem "přirozeně se vyskytující text" je míněn text v jakémkoliv jazyce, jakéhokoliv stylu či žánru, který může být jak psaný, tak podán ústním projevem. Jediným omezením je, aby text reprezentoval mezilidskou komunikaci a aby nebyl speciálně vytvořen pro účely přirozeného zpracování řeči, ale aby byl spíše výsledkem skutečného užití. Lidé používají několik úrovní lingvistické analýzy pro stavbu přirozeného textu, na rozdíl od lidí mohou ale programy využívat pouze některé úrovně, či jejich kombinace.

6.1 Lexikální a syntaktická analýza

Lexikální analýza se zabývá smyslem individuálních slov. Základem lexikální analýzy je identifikování slovních druhů ve větě. Úroveň syntaktické analýzy zkoumá gramatiku vět. K takovému zkoumání pořebujeme *gramatiku* a *parser*. Výstupem této analýzy může být nelineární struktura reprezentující strukturu a závislosti mezi slovy. Součástí syntaktické analýzy je také odstranění nejednoznačností u slov, které mají zastoupení ve více slovních druzích.

6.2 Stanford Log-linear Part-Of-Speech Tagger

Part-Of-Speech Tagger[15] je knihovna, které dokáže identifikovat slovní druhy ve větách textu. Výsledný text je označkován pomocí tagů, které jsou popsány v [21]. Tato knihovna bude použita při implementaci indexu *nálady trhu* viz 7.2.

6.3 Sentimentální analýza

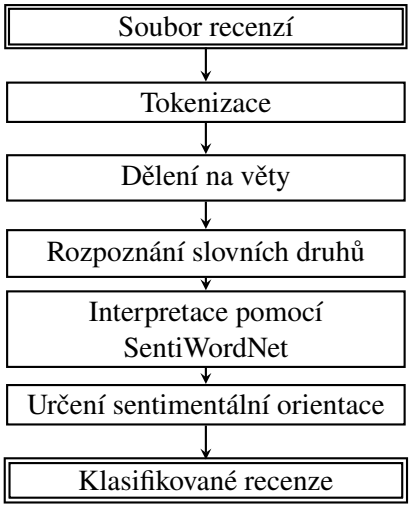
Pro tuto práci je sentimentální analýza velmi důležitá, protože právě sentimentální analýza bude tvořit základ navrženého indexu *nálady trhu*. Podle [16] používá sentimentální analýza zpracování přirozené řeči, statistiku nebo strojové učení pro extrahování, identifikování nebo charakterizování sentimentu v textu. Sentiment je možný postoj, emoce nebo názory, jejichž polaritu se snažíme odhalit. Sentimentální analýza může být využita k hledání odpovědí z recenzí, internetových zpráv, popřípadě z jakéhokoliv textu, který se jasně týká specifického problému. Můžeme hledat odpovědi na otázky ohledně entit jako jsou filmy, knihy, auta, mobilní telefony, společnosti atp. Sentimentální analýza (i jiné typy analýz) je využita v systému AzFinText, jak je uvedeno v podkapitole 3.3.

6.4 Senti Word Net

Podle [18] je SentiWordNet nástroj pro sentimentální analýzu. SentiWordNet pracuje se *sensety*[16], kde každý synset má přiřazen slovní druh a je ohodnocen třemi hodnotami:

[15] *Tag* znamená označit, nebo značka; http://nlp.stanford.edu/software/tagger.shtml

[16] *Synset* je skupina synonym



Obrázek 6: Proces kategorizace recenzí podle sentimentální analýzy

- pozitivitou - číslo od nuly do jedné; znamená, jak moc je *synset* pozitivní,
- negativitou - číslo od nuly do jedné; znamená, jak moc je *synset* negativní,
- objektivitou - jedná se pouze o hodnotu vypočtenou z předchozích dvou následujícím vzorcem: 1 - (pozitivita + negativita)

Autor [19] tvrdí, že Senti Work Net lze využít pro dolování sentimentu a kategorizaci textu tak, jak je zobrazeno na obrázku 6. Nejdříve je nutné mít k dispozici soubory textu, poté prochází text procesem zpracování. Důležitým bodem je *Speech Tagging*, k čemuž lze využít například *Stanford Log-linear Part-Of-Speech Tagger*, viz 6.2. Následuje vyhledání záznamu v databázi *Senti Word Net*, výpočet sémantické orientace a klasifikace do tříd.

7 Implementace simulace automatického obchodního systému

Motivací pro implementaci simulace automatického obchodního systému je několik.

- simulované strategie lze použít při ostrém obchodování v kombinaci s některými stávajícími systémy uvedenými v kapitole 3,
- vlastní řešení je možné optimalizovat co se týče rychlosti libovolně, což může pravděpodobně vést k rychlejší odezvě systému,
- implementací vlastního řešení získá autor znalost dané problematiky.

7.1 Data

Pro simulaci obchodování potřebujeme přístup k historickým datům. Data jsou přístupná na internetu v různých formátech. Implementovaný simulační program umožňuje použít dva zdroje dat, těmi jsou Yahoo Finance API a Google Finance API. Yahoo Finance API poskytuje data s denní přesností a Google Finance API poskytuje data až s minutovou přesností, avšak ne po celý den. Pro přístup ke zprávám o akciovém trhu používá simulační program stránku www.finviz.com, která zobrazuje poslední zprávy o akciových titulech.

7.1.1 Yahoo Finance API

Yahoo Finance API je volně dostupný zdroj historických burzovních dat. Informace o API byly čerpány z *Yahoo Finance Managed* [17]

```
http :// ichart .yahoo.com/table.csv?s=[TICKER]
```

Výpis 1: Šablona URL pro přístup k datům - Yahoo Finance API

Tato práce využívá část API poskytující data ve formátu *csv*[18], který má následující strukturu: Na prvním řádku jsou uvedeny významy hodnot ve sloupích počínaje datem, poté údaji o ceně, objemu atd. Formát je ukázán ve výpisu 2. Příklad URL pro přístup k datům je ukázán v 1. Nahrazením řetězce **[TICKER]**[19] za skutečnou hodnotu získáme použitelnou adresu.

```
Date,Open,High,Low,Close,Volume,Adj Close
2014-04-22,528.31,531.83,526.50,531.70,7234400,531.70
2014-04-21,525.34,532.14,523.96,531.17,6519600,531.17
2014-04-17,520.00,527.76,519.20,524.94,10154800,524.94
2014-04-16,518.05,521.09,514.14,519.01,7670200,519.01
```

Výpis 2: Yahoo Finance API - formát navrácených dat

[17] `https://code.google.com/p/yahoo-finance-managed/` obsahuje detailní popis všech funkcí rozhraní, spolu s jednoduchou implementací a ukázkou, jak API používat.

[18] *Comma Separated Values*, hodnoty jsou odděleny čárkou, popřípadě jiným znakem; jednotlivé záznamy jsou odděleny řádky

[19] Ticker je burzovní symbol označující cenné papíry

7.1.2 Google Finance API

Další alternativou pro získání historických dat je Google Finance API. Jedná se o nedokumentované API, které nabízí historické ceny akcií s minutovou přesností. Formát výstupu není nikde oficiálně zdokumentován, respektive dokumentaci nebylo možné vyhledat, pokud existuje. Na internetových fórech je formát popsán velmi přibližně, proto bylo nutné experimentovat. Z experimentování také vyplynulo, že toto API poskytuje data jen pro čas 14:30 až 21:00 UTC, což odpovídá otevírací době burze NYSE a NASDAQ v New Yorku.

```
https :// www.google.com/finance/getprices?i=[INTERVAL]&p=[DNY]d&f=d,o,h,l,c,v&q=[TICKER]
```

Výpis 3: Šablona URL pro přístup k datům - Google Finance API

Výpis 3 obsahuje šablonu, pro přístup k datům. Na místo řetězce **[INTERVAL]** se dosazuje počet sekund. Minimální hodnotou je 60 sekund. Pokud uvedete hodnotu, která je menší, například 59, bude každý řádek odpovědi obsahovat časové razítko. Pokud tak neučiníte a zadáte například 60, bude časové razítko uvedeno jen na prvním řádku odpovídajícím časové periodě otevírací doby burzy v New Yourku.

Číselná hodnota, kterou zaměníme za řetězec **[DNY]** značí, kolik dní nazpět má být navráceno od aktuálního momentu. API umožňuje specifikování hodnoty v řádech let, ale pro naše účely potřebujeme specifikovat dny.

Posledním parametrem, který je nutné specifikovat zaměněním řetězce **[TICKER]**, je označení titulu, jehož historická data požadujeme. Zde je možné vyplnit hodnoty jako “GOOG”, “AAPL” atp[20]. Za zmínku stojí část URL *f=d,o,h,l,c,v*, kde první parametr, označený písmenem *f*, je výstupní formát dat. Hodnoty jsou akronymy pro date (datum), open (otevírací cena), high (nejvyšší cena), low (nejnižší cena), close (uzavírací cena), volume (objem transakcí). Formát pro naše účely zůstává neměnný. Formát navrácených dat je ukázán ve výpisu 5. Kompletní ukázka URL pro přístup k datům je ve výpisu 4

```
https :// www.google.com/finance/getprices?i=59&p=3d&f=d,o,h,l,c,v&q=GOOG
```

Výpis 4: Ukázkový příklad URL pro přístup k historickým cenám akcií Googlu

```
EXCHANGE%3DNASDAQ
MARKET_OPEN_MINUTE=570
MARKET_CLOSE_MINUTE=960
INTERVAL=59
COLUMNS=DATE,CLOSE,HIGH,LOW,OPEN,VOLUME
DATA=
TIMEZONE_OFFSET=-300
a1392820200,1204.11,1205.6,1204.11,1205.3,37744
a1392820260,1206.135,1206.57,1203.15,1204.715,24794
a1392820320,1205.16,1205.39,1204.241,1204.28,9565
```

Výpis 5: Google Finance API - formát navrácených dat

[20] Seznam všech významných *tickerů* je možné stáhnout zde `http://finviz.com/export.ashx?f=idx_sp500`

7.1.3 Finviz

V této práci je nastínem způsob obchodování na základě analýzy textů zpráv. Pro tuto analýzu je tedy nutné automatizovat přístup k potřebným datům. Internetová stránka www.finviz.com udržuje seznam relevantních zpráv (odkazů) ke každému akciovému titulu. Odkazy na internetové články je možno stáhnout zadáním dotazu, jak je vidět ve výpisu 6 - řetězec **[TICKER]** musíme zaměnit za skutečnou hodnotu. Odpověď obsahuje spoustu informací, která pro nás v tuto chvíli není podstatná. Za použití knihovny *Jsoup*[21] je možné jednoduše odfiltrovat odkazy, které potřebujeme. Tyto odkazy identifikují podstatné zprávy týkající se akciového titulu.

```
http :// finviz .com/quote.ashx?t=[TICKER]
```

Výpis 6: Šablona URL pro stažení stránky s relevantními informacemi o akciovém titulu

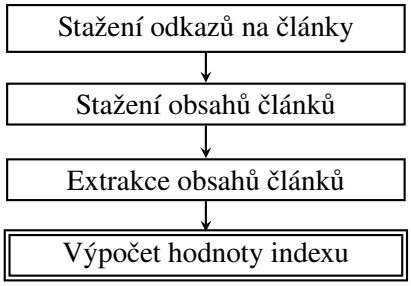
Nyní máme k dispozici hypertextové odkazy, tedy kolekci URL elementů a obsah, na který odkazy ukazují, může být jednoduše stažen. Stažení obsahu článků je nutné implementovat konkurentně, protože stahování velkého počtu zpráv se tím výrazně urychlí. Při práci s vlákny je velmi výhodné použití knihovny *Guava* [22], která nabízí podstatné ulehčení i pro jiné rutinní operace. Pro samotnou HTTP komunikaci není vhodné používat třídu URL, ale je vhodné použít robustnější knihovnu, například *Apache HttpComponents* [23], která je výkonnější. V době psaní části programu, která má na starosti ono stahování zpráv, byla aktuální verze JDK 7, a proto bylo výhodné použít knihovnu Guava. V nově vydaném JDK 8 se naskýtá řešení konkurentního stahování pomocí metody parallelStream(), což by mohlo vést až k odstranění závislosti na knihovně třetí strany.

Když jsou články k vybranému akciovému titulu staženy, čelíme podstatnému problému, tím je neuniformní formát zpráv na každém zpravodajském serveru. Stránka obsahující zprávu s sebou nese spoustu nevyžádaných informací, jako jsou internetové reklamy apod. Tento problém se dá vyřešit použitím knihovny *Boilerpipe* [24], což je knihovna, která umí s dobrou přesností extrahovat pouze relevantní data z webové stránky a je vhodná právě pro webové stránky, obsahující zprávy. V případě, že algoritmus knihovny neextrahuje text zprávy správně, je možné implementovat extrakci zpráv pro daný zpravodajský server zvlášť pomocí knihovny *Jsoup*, takže je možné do budoucna zvýšit přesnost extrakce textu zprávy.

7.2 Implementace indexu *nálady trhu*

V této části práce je popsán proces získání hodnot indexu "nálady trhu" a způsob, jakým se tento index dá použít v obchodování. Základní myšlenkou je získání sentimentální orientace (polarity) z internetových textů, kdy ke každému textu musíme znát i čas uveřejnění, abychom mohli index vůbec vykreslit do grafu. Index je možné sledovat například pro celý trh s akciemi, nebo jenom pro určitou část. V této práci jsem se zaměřil na jednotlivé akciové tituly, spíše než na celý trh, protože právě nálada daného akciového titulu nám může usnadnit obchodování s jeho akciemi.

[21] http://jsoup.org/
[22] https://code.google.com/p/guava-libraries/
[23] https://hc.apache.org/
[24] https://code.google.com/p/boilerpipe/



Obrázek 7: Proces výpočtu indexu pro zadaný ticker

7.2.1 Výpočet hodnoty indexu *nálady trhu*

Pokud již máme k dispozici holý text zprávy, můžeme aplikovat metody zpracování přirozeného jazyka. Výpočet hodnoty indexu probíhá v několika krocích. Prvním krokem je provedení lexikální a syntaktické analýzy, druhým krokem je provedení sentimentální analýzy se samotným výpočtem indexu. K lexikální a syntaktické analýze textů jsem použil *Stanford Log-linear Part-Of-Speech Tagger* [25]. Určení slovních druhů je podstatné pro další krok, ve kterém se slovo vyhledá v databázi *SentiWordNet*, která je popsaná v kapitole 6. Výpočet indexu je inspirován článkem o kategorizaci recenzí podle sémantické orientace, jak uvádí [19].

Výpočet tedy podrobí text finanční zprávy lexikální analýze a pokusí se vyhledat synset podle slova a jeho slovního druhu v databázi *SentiWordNet*. Pokud synset nalezen není, pokračujeme dalším slovem. V případě opačném, je nalezen jeden, nebo více synsetů a my z nich musíme určit sentimet původního slova. Zde nastává problém, protože nejsme schopni jednoznačně vybrat synset podle smyslu slova, proto se zde dá zavést zjednodušení a to takové, že uděláme průměr hodnot všech nalezených synsetů. Sentimentální orientaci synsetu (*S*) vypočítáme pomocí vzorce 3, kde *p* pozitivita synsetu, *n* je negativita synsetu. Závorka udává pouhý rozdíl mezi hodnotou pozitivity a negativity. Udává, o kolik je synset více či méně pozitivní, než negativita.

$$S = (p - n) \tag{3}$$

Sentimentální orientace textu je pak spočítána jako průměrná hodnota všech slov. Celý výpočet sentimentální orientace textu jednoho článku (S_p) je znázorněn vzorcem 4, kde p_s je počet slov v článku, p_n je počet nalezených synsetů na základě slova a jeho slovního druhu, p_{ij} je pozitivita synsetu a n_{ij} je negativita synsetu.

$$S_p = \frac{\sum_{i=1}^{p_s} \frac{\sum_{j=1}^{p_n} (p_{ij} - n_{ij})}{p_n}}{p_s} \tag{4}$$

[25] `http://nlp.stanford.edu/software/tagger.shtml` a kapitola 6

7.2.2 Obchodování pomocí indexu

Je důležité si uvědomit, že zprávy nepřicházejí v čase pravidelně. Pokud chceme hodnotu indexu využívat simultálně, musíme se zamyslet, jak hodnoty indexu interpolovat pro jiné časové body. První možnost je triviální a to taková, že každá příchozí nová hodnota indexu zůstane platná, dokud nepřijde další hodnota. Druhou možností je využití nějakého technického indikátoru, například SMA či EMA, což by mělo za následek zprůměrování indexu. Protože jsou zprávy a jejich příchod nepravidelný, bylo by možno pracovat s indexem jako s váženým průměrem. Čím starší by zpráva byla, tím by informace ztrácela na důležitosti.

V této práci jsem se zabýval možností, ve které využívám příchod nové zprávy jako signál (událost), že má proběhnout rozhodnutí o nákupu či prodeji. Strategie je tedy aktivována, jakmile je detekována existence nové zprávy a hodnota vypočítáného indexu se spolu využije k predikci ceny a k potenciálnímu nákupu či prodeji.

7.3 Simulační model

Pro simulaci obchodování bylo nutné implementovat celý systém komponent, které představují simulační model. Každá komponenta přebírá v simulaci zodpovědnost za různé činnosti a je zjednodušením reálného světa. V této podkapitole jsou komponenty popsány detailně s důrazem na požadavky, které splňují. Simulační model je realizovaný v jazyce Java. Pro simulování je velmi výhodné pracovat s objektově orientovaným návrhem.

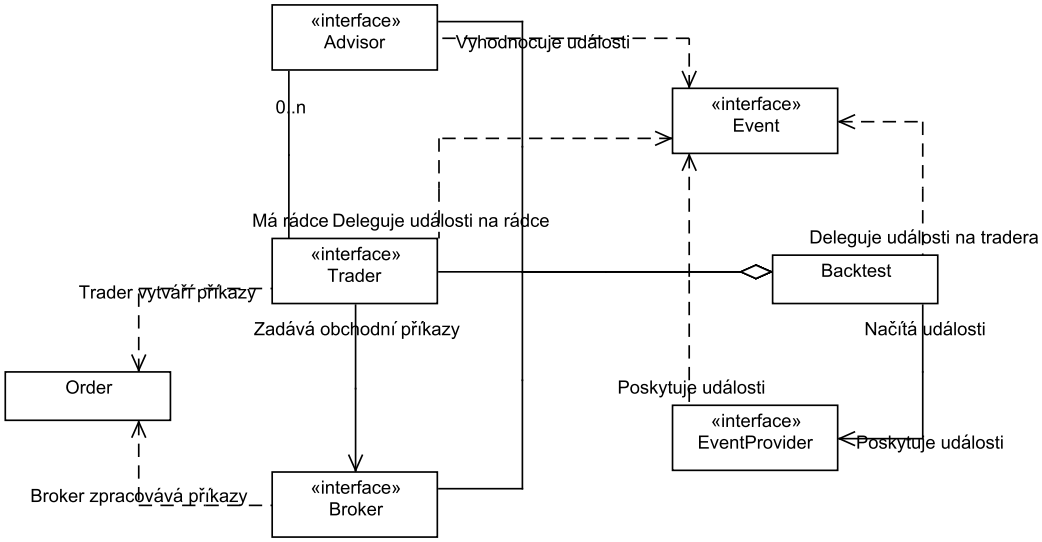
7.3.1 Diagram tříd hlavních komponent

Obrázek 8 znázorňuje diagram tříd hlavních komponent. Vedlejší nedůležité třídy, z celkového počtu více než 170 tříd[26], byly vynechány, aby byla zachována přehlednost a aby vynikla hlavní myšlenka modelu.

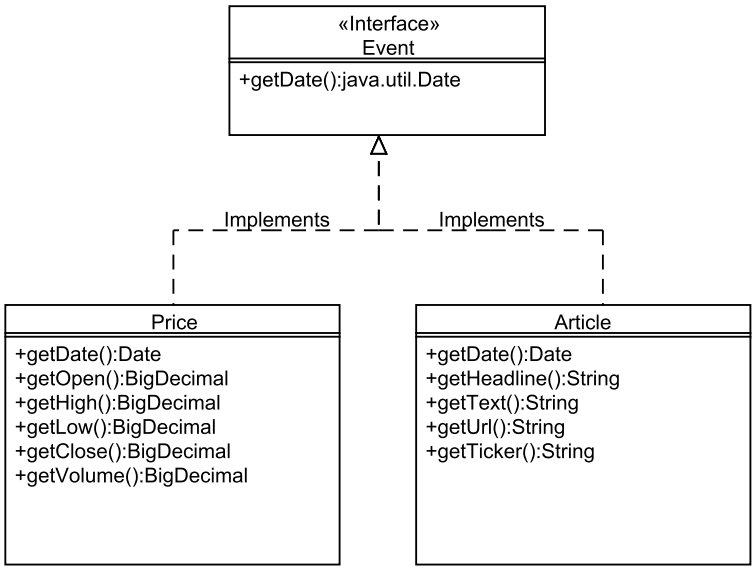
7.3.2 Event

Událost může v simulaci reprezentovat dvě možné situace, první situací je příchod finanční zprávy a druhou situací je příchod informace o ceně. Události jsou hlavním vstupem pro obchodní strategii. Pokud je událost typu Price (Cena), obsahuje událost kromě datumu a času hodnoty otevírací, nejvyšší, nejnižší a uzacírací ceny a hodnotu obchodovaného objemu, pokud dostupný zdroj objem poskytuje. Strategie implementované v této práci využívají pouze datum, čas a uzavírací cenu. Pokud je událost typu Article (článek), obsahuje kromě datumu a času i text samotného článku, titulek, zdrojové url a ticker. Stejně jako u události typu Price, používají strategie implementované v této práci pouze některá data, těmi jsou: datumu a času a text článku. Ostatní informace mohou být využity při rozšíření programu. Události jsou dostupné pomocí rozhraní EventProvider. Na obrázku 9 je znázorněn třídní diagram událostí.

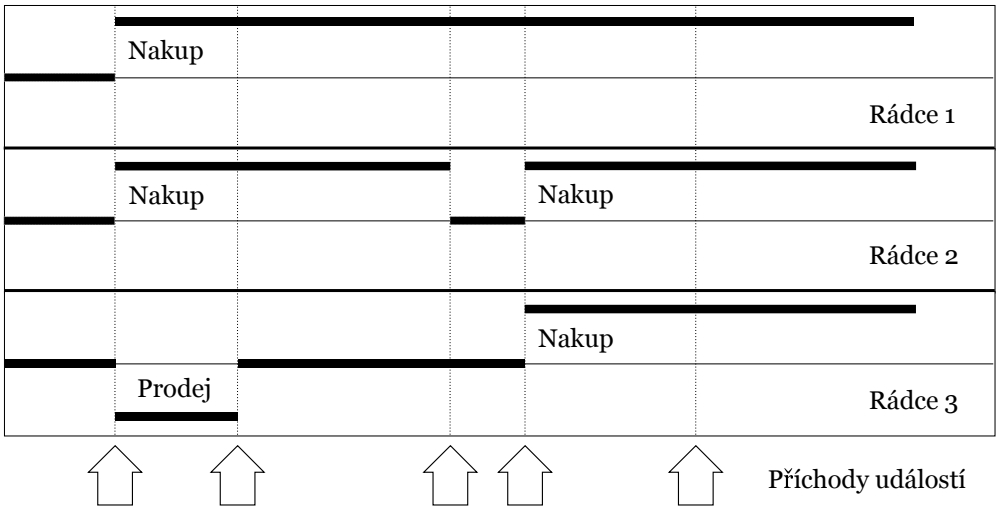
[26] Ne všechny třídy jsou součástí simulačního modelu. Jedná se i o utilitní třídy, abstraktní třídy, rozhraní, enumerátory atp.



Obrázek 8: Diagram tříd hlavních komponent



Obrázek 9: Třídní diagram znázorňující události detailně



Obrázek 10: Příchody událostí a reakce rádců

7.3.3 Event Provider

Tato komponenta zpřístupňuje v simulaci historická data. Její specifické implementace mají na starosti stažení dat z API Yahoo, Googlu a zpráv ze serveru Finviz. Data se hromadně stáhnou na začátku simulace. Při optimalizaci zvoleným optimalizačním algoritmem jsou data znovu používána a nedochází k opakovanému stahování stejných dat stále dokola.

7.3.4 Trader

Trader reprezentuje entitu obchodníka, který sleduje grafy a zadává obchodní příkazy. V této simulaci zastupuje fyzického obchodníka automatický obchodník, stejně jako je to běžné u automatických obchodních systémů, ve kterých se lidský faktor na obchodování nepodílí (neměl by). Vstupem pro obchodníka jsou ceny na burze, popřípadě finanční zprávy a výstupem jsou objednávky pro vstup do pozice. Signály pro vstup nevyhodnocuje komponenta sama, ale využívá rádců, viz Advisor. V naší simulaci bude obchodník obchodovat pouze s objemem, který bude na začátku simulace odvozen od ceny akcie simulovaného tickeru pro hodnotu 5000 USD[27].

7.3.5 Order

Order je objekt sloužící jako zpráva od *obchodníka brokerovi*. Obsahuje údaj o požadovaném objemu. Kladný objem znamená nákup a záporný znamená prodej. Dále obsahuje hodnotu *stop loss* a *take profit*[28]

[27] Hodnota není nastavitelná, vhodné pro rozšíření

[28] *Take profit* se momentálně nepoužívá, vhodné pro rozšíření.

7.3.6 Advisor

Advisor (rádce) má za úkol reagovat na změnu ceny či příchod finanční zprávy a detekovat, kdy je pro obchodníka dobré do trhu vstupovat, kdy by neměl obchodník reagovat vůbec a nakonec, kdy by měl z pozice vystoupit. Obchodník může při svém obchodování používat více rádců a vstupovat do obchodů pouze tehdy, pokud je to pro něj bezpečné, tedy pokud se všichni rádci shodují a naopak vystoupit z obchodu v situaci, kdy libovolný rádce výstup z obchodu doporučí. Na obrázku 10 je znázorněna situace, kdy po příchodu čtvrté události vyhodnotí automatický obchodník, že má vstoupit do pozice. Kdyby poté jediný rádce doporučil z pozice vystoupit, obchodník by tak reagoval. Tato strategie je v modelu pevně daná. Implementace více strategií v rozhodnování je jedno z možných vylepšení do budoucna. Bylo by například vhodné, kdyby se logika pro vstupování a logika pro vystupování z pozic oddělila a obchodník by využíval pro každou činnost jiné rádce, je totiž možné (hypoteticky), že by například neuronová síť dobře fungovala jako rádce pro vstup do pozice a naopak nějaký jiný rádce by byl lepší pro vystupování z pozic.

SMA Advisor využívá indikátoru SMA, který se používá na burze jako indikátor hlavního trendu. Hodnota indikátoru je průměrem posledních n hodnot zavírací ceny. Strategie založená na SMA sleduje, kdy se hlavní trend protne s aktuální cenou. Takový moment znamená signál k nákupu či prodeji. Pokud aktuální cena protíná hlavní trend směrem dolů, znamená to signál k prodeji a naopak. Jediným optimalizovatelným parametrem této komponenty je perioda. Indikátor je popsán v 2.2.1.

RSI Advisor je založen na hodnotách indikátoru RSI. Optimalizovatelnými parametery komponenty jsou: perioda, hranice překoupenosti a hranice přeprodanosti.

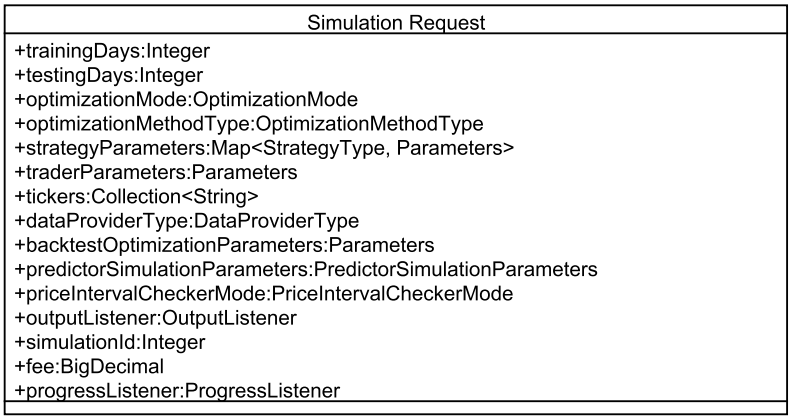
Dalším typem rádce je Predictor Advisor, což je rádce založený na predikování budoucích hodnot z hodnot předchozích. K predikci používá rádce neuronovou síť. Kromě optimalizace neuronové sítě je rádce řízen horním a dolním hraničním parametrem. Pokud je predikovaná hodnota vyšší než horní mez, doporučuje rádce kupovat, pokud je nižší než mez spodní, doporučuje prodávat, jinak nedoporučuje nic. Tato část strategie využívá framework Encog[29].

Posledním typem rádce je Mood Advisor, což je rádce pracující s indexem *nálady trhu*, který byl v rámci této práce navržen a implementován viz 7.2. Strategie rádce je naprosto stejná, jako u předchozího rádce, s tím rozdílem, že hodnota, podle které probíhá rozhodování o nákupu a prodeji, je založena na hodnotě indexu získané z internetových finančních zpráv. Každý příchod *zprávy* může znamenat rozhodnutí o nákupu. Mezi dvěma příchody události typu *cena* může přijít několik *zpráv* a na každou zprávu může obchodník reagovat, což je výhoda, oproti jiným rádcům, kteří pouze z ceny těžce určí, jaký objem mají nakoupit.

7.3.7 Backtest Broker

Broker má na starosti provedení obchodů obchodníka. Disponuje také informací o aktuálním stavu účtu obchodníka. Při simulaci je klíčové správně vystihnout chování brokera tak, jak by se choval při opravdovém obchodování. Broker si strhává poplatky za provedení transakce. Poplatek za provedení transakce je pro simulaci konstantní a dá se nastavit jako parametr. Broker také automaticky kontroluje automatické vystoupení z pozice pomocí *stop loss*, což je parameter, který je možné v simulaci optimalizovat.

[29] `http://www.heatonresearch.com/encog`



Obrázek 11: Simulační požadavek

7.3.8 Backtest

Komponenta Backtest agreguje všechny předchozí komponenty a stará se o řízení simulace. Je to řídící komponenta, která iteruje událostmi v časovém sledu a zasílá události komponentám Trader a BacktestBroker, které jsou tak pokaždé synchronizovány.

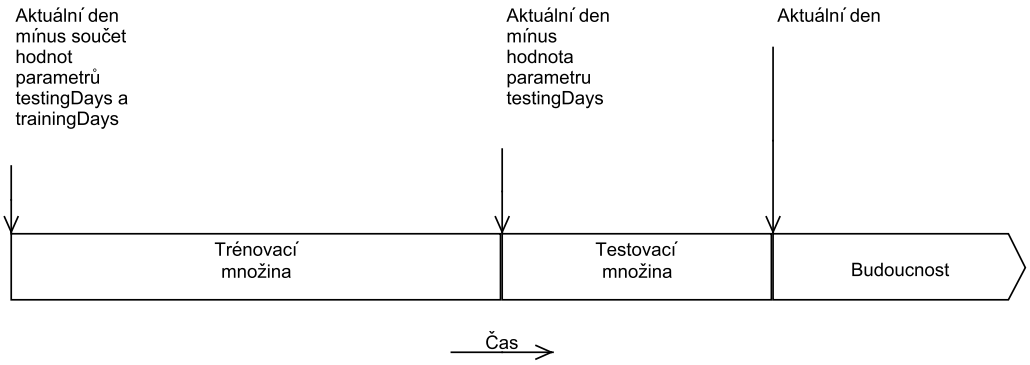
7.3.9 Backtest Solution

Abstraktní komponenta, jedná se o třídu, která při optimalizaci reprezentuje jedince. Třída implementuje rozhraní *Solution*, což je funkční rozhraní, které je využito v knihovně jExtreme, viz kapitola 7.5, a tím zaobaluje simulaci obchodování jednoho akciového titulu a umožňuje zmíněnou optimalizaci.

7.4 Proces simulace

Celý proces simulace a optimalizace strategií je implementován formou služby, což z architektonického hlediska umožňuje budoucí přepsání uživatelského rozhraní do jakékolik prezentační technologie, například do HTML 5, popřípadě ze simulační služby udělat nástroj příkazového řádku, nebo využívat simulační službu přes webovou službu.

Na vstupu simulační služby je simulační požadavek, který je znázorněn na obrázku 11. Parametrem **testingDays** definujeme počet dnů pro data testovací množiny. Pokud má parametr hodnotu 3, znamená to, že bude testovací množina obsahovat data za poslední tři dny. Obdobou parametru testingDays je parametr **trainingDays**, který značí počet dnů pro trénovací množiny. Testovací a trénovací množina jsou disjunktní množiny dat. Situace je znázorněna na obrázku 12. V případě, že je použit index *nálady trhu*, jsou hodnoty parametrů ignorovány. Délky testovací a trénovací množiny jsou vypočítány automaticky pro všechny dostupné zprávy a jsou symetricky rozděleny na poloviny.



Obrázek 12: Trénovací a testovací množina

Typ rádce (AdvisorType)	Vlastnost	Výchozí hodnota	Minimální hodnota	Maximální hodnota
SMA	period	30	2	100
RSI	period	14	1	100
RSI	overbought	70	50	100
RSI	oversold	30	0	50
Predikce ceny	positiveDelta	2	0	20
Predikce ceny	negativeDelta	2	0	20
Index nálady trhu	positiveMoodBoudary	0.005	0	1
Index nálady trhu	negativeMoodBoudary	0.005	0	1

Tabulka 1: Přehled rádců a jejich parametrů

Hodnota parametru **optimizationMode** umožňuje vypnout či zapnout optimalizaci. Možné hodnoty jsou pouze ON a MANUAL. Pokud chceme obchodní strategii optimalizovat, musíme vybrat optimalizační metody pomocí parametru **optimizationMethodType**. Možné hodnoty jsou BLIND_ALGORITHM, GENETIC_ALGORITHM_OWN, SOMA.

Parametr s názvem **strategyParameters** obsahuje nastavení rádců (7.3.6). Parametr je typu mapa a v případě, že chceme využít optimalizaci, budou použity pouze klíče mapy a jejich hodnoty budou zahozeny. V opačném případě, když je optimalizace vypnutá, se použijí jak klíče, tak hodnoty. Současná implementace zavádí omezení a to takové, že nelze specifikovat použití více rádců stejného typu s jinými parametry. Nelze například vytvořit strategii využívající dva rádce pracující s indikátorem SMA, kde každý rádce využívá jinou hodnotu periody indikátoru. Toto omezení není pro cíl této práce limitující, ale bylo by dobré implementaci změnit. Všechny hodnoty, které je možné specifikovat, jsou uvedeny v tabulce 1. Všechny datové typy parametrů jsou *java.lang.Double*. Všechny parametry mohou být nakonfigurovány v souboru *strategy-configuration.csv*

Obdobou parametru strategyParameters je parameter **traderParameters**. Jedná se o parametry

Vlastnost	Výchozí hodnota	Minimální hodnota	Maximální hodnota
stopLossPips	20	1	100

Tabulka 2: Parametr automatického obchodníka

Algoritmus	Parametr	Výchozí hodnota	Typ
Slepý algoritmus	Počet kroků	10000	Long
Genetický algoritmus	Velikost populace	100	Integer
Genetický algoritmus	Počet kroků (generací)	1000	Long
Genetický algoritmus	Pravděpodobnost mutace	0.2	Double
Genetický algoritmus	Poměr elitismu	0.05	Double
SOMA	Počet kroků (migrací)	400	Long
SOMA	Velikost populace	20	Integer
SOMA	PRT	0.1	Double
SOMA	Krok	0.099	Double
SOMA	Délka cesty (Path Length)	3	Double

Tabulka 3: Parametry optimalizačních algoritmů

obchodníka. Momentálně existuje pouze jedinný parametr komponenty Trader, a tím je stopLoss, což je hodnota, který udává maximální možný pokles akcie, jakmile klesne cena pod tuto hodnotu, broker automaticky akcie prodá. Parametr je popsán v tabulce 2. Parametry mohou být upraveny v souboru *trader-configuration.csv*.

Parametr **tickers** obsahuje kolekci řetězců reprezentujících jednotlivé akciové tituly. Simulace a optimalizace, pokud ji vyžadujeme (viz předchozí parametery optimizationMode a optimizationMethodType), proběhne pro každý z uvedených tickerů paralelně. Simulační služba využívá volání parallelStream() pro realizaci konkurence. Chyba při simulaci tickeru neovlivní simulaci ostatních a chyba bude zaznamenána. Běžným problémem, který vzniká při simulaci většího počtu tickerů zároveň, je odmítnutí poskytnutí historických cen pod stavovým kódem HTTP 502. V takovém případě nezbývá nic jiného, než proces pro daný ticker opakovat.

Parametrem **dataProviderType** je možné zvolit server, ze kterého se načtou data historických cen, viz 7.1. Je možné vybrat ze dvou hodnot, a těmi jsou GOOGLE a YAHOO.

Hodnoty nastavení optimalizace se předávají do simulačního procesu pomocí parametru **backtestOptimizationParameters**. Specifikujeme velikost populace, počet evolučních kroků atp. v závislosti na zvolené metodě optimalizace. Přehled parametrů je v tabulce 3. Parametry mohou být upraveny v souboru *optimization-configuration.csv*.

Parametry pro vytvoření prediktoru jsou specifikovány parametrem **predictorSimulationParameters**. V našem případě je jediným možným typem prediktoru neuronová síť, takže parametry obsahují informace nutné pro vytvoření a natrénování neuronové sítě. Všechny parametery jsou zobrazeny v tabulce 4. Parametry mohou být upraveny v souboru *nn-predictor-configuration.csv*.

priceIntervalCheckerMode umožňuje zapnout tzn. kontrolu intervalu ceny. Tím se myslí, že použité indikátory a rádci, kteří je využívají, reagují pouze tehdy, pokud je dodržena kontinuita

Parametr	Výchozí hodnota	Typ
Perioda	5	Integer
Počet skrytých vrstev	2	Integer
Error	0.001	Double
Maximální počet iterací	100000	Integer
Velikost testovací množiny	30	Integer
Počet iterací pro detekci uvíznutí	10000	Integer
Epsilon pro detekci uvíznutí	0.0000000001	Double
Transformace vstupů	1.5	Double
Chytré učení	true	Boolean
Learning Rate	0.7	Double
Momentum	0.2	Double

Tabulka 4: Parametry neuronové sítě.

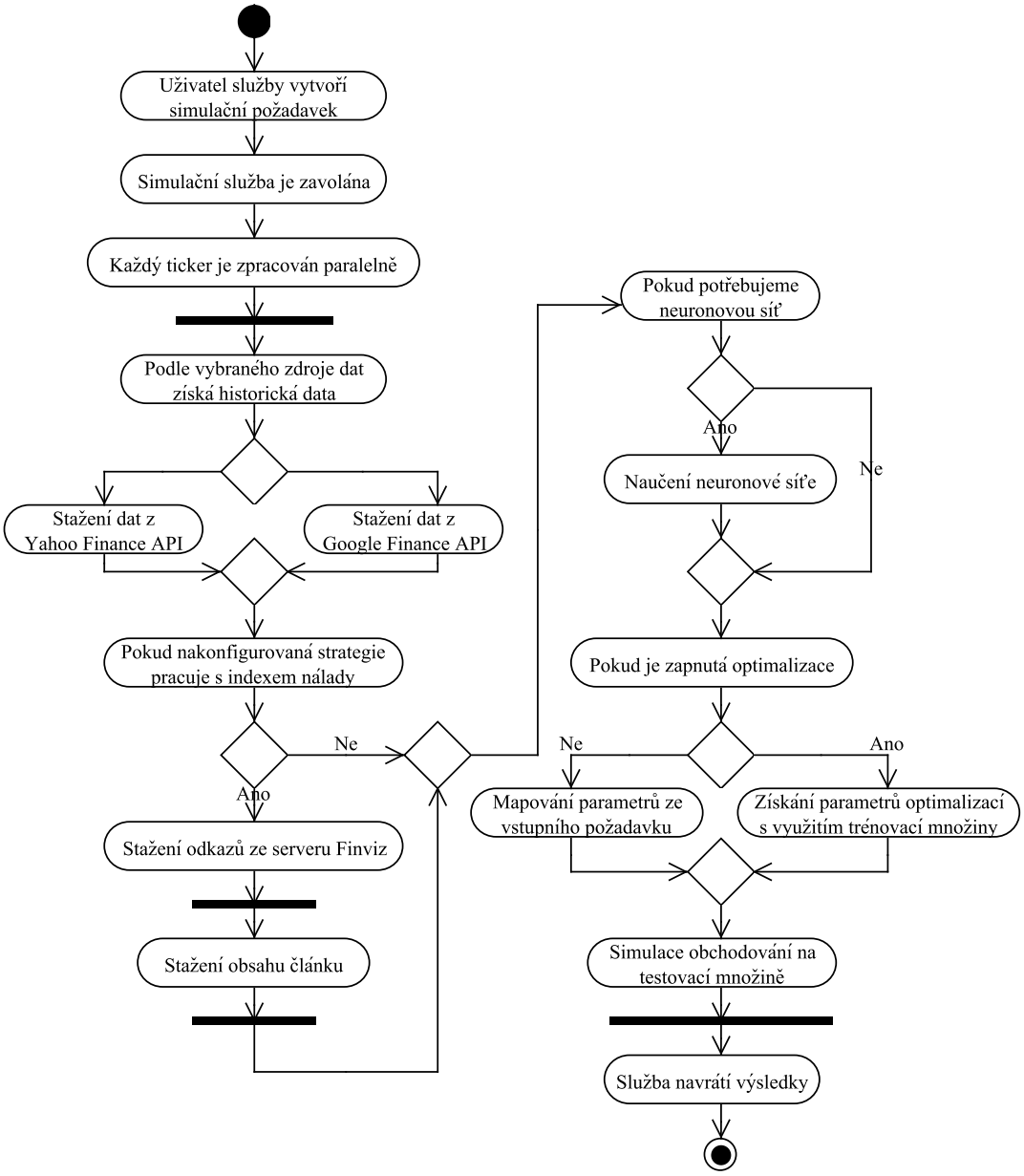
dat. Například nezareagují tehdy, pokud používáme denní hodnoty z Yahoo Finance API a zároveň indikátor SMA s periodou 6. API poskytuje hodnoty pouze pro všední dny, a proto nikdy nenastane situace, kdy je kontinuita dodržena po celou periodu indikátoru SMA.

outputListener - jako parameter může být předán objekt, který bude sloužit pro naslouchání textových výstupů simulace. Výstupy mohou být zobrazeny v konzoli nebo ukládány do souboru, dle potřeby simulationId Korelační identifikátor. Je předán zpět do objektu s výsledkem. Hodí se především v situacích, kdy spouštíme více simulací zároveň.

fee je číselný parametr, který určuje velikost poplatku za jednu provedenou transakci. Parametr je při simulaci konstantní, takže se jedná o zjednodušení. Výše poplatku, která odpovídá výši reálného poplatku, je přibližně hodnota 10.

progressListener Jako parametr může být předán objekt, který bude sloužit pro monitorování stavu. Přístup je podobný jako u parametru outputListener, nejedná se ovšem o naslouchání textových dat, ale o sledování stavu simulace.

Proces simulace je znázorněn na obrázku 13. Proces začíná vytvořením objektu simulačního požadavku a následným zavoláním simulační služby. Simulační služba poté paralelně zpracovává každý ticker, následující část procesu tedy probíhá identicky několikrát pro každý ticker. Jsou stažena historická data (ceny) z požadovaného zdroje. Pokud strategie bude využívat index *nálady trhu*, jsou staženy odkazy ze serveru Finviz a poté jsou paralelně staženy obsahy článků. V případě, že potřebujeme neuronovou síť, tedy v tom případě, kdy budeme provádět predikci budoucích cen z cen minulých, musíme síť naučit na trénovací množině. Pokud je zapnutá optimalizace, jsou parametry optimalizovány pomocí vybraného algoritmu na trénovací množině. Pokud je optimalizace vypnutá, namapují se parametry ze simulačního požadavku tak, jak je zadal uživatel. Posledním krokem pro ticker je simulace na testovací množině. Jakmile jsou všechny tickery konkurentně odsimulovány, navrátí služba hromadně všechny výsledky.



Obrázek 13: Proces simulace znázorněný pomocí aktivitního diagramu

Obrázek 14: Dynamická tvorba specimenu

7.5 Optimalizace a optimalizační knihovna jExtreme

Pro účel optimalizace obchodních strategií byla v rámci práce vyvinuta knihovna *jExtreme*[30] v jazyce Java 8, která obsahuje implementace tří optimalizačních algoritmů. Prvním algoritmem je "Blind Algorithm (slepý algoritmus)", což je nejjednodušší optimalizační algoritmus, který pseudonáhodně prohledává prostor možných řešení. Druhým algoritmem je "genetický algoritmus", který na rozdíl od běžného přístupu nepracuje s jedinci v binárním kódu, ale s jedinci, jejichž genotyp je uložen v kolekci hodnot typu Double. Třetím algoritmem je "Self-Organized Migration Algorithm" (Samo-organizující se migrační algoritmus, zkráceně SOMA).

7.5.1 Specimen

Je důležité si uvědomit, jaké parametry se při simulaci optimalizují. Simulace obsahuje velké množství řídících parametrů a pro čtenáře by toto mohlo být zavádějící. Parametry, které jsou optimalizovány v rámci simulace, jsou uvedeny v tabulkách 1 a 2 a záleží na uživateli, jak nakonfiguruje strategii, protože podle toho se mění optimalizační úloha. Strategie se pokaždé skládá z množiny rádců a automatického obchodníka a právě to, jaké rádce uživatel specifikuje, bude mít vliv na tvorbu *specimenu*. Na obrázku 14 je znázorněno, jak je specimen uspořádán. Nejdřív jsou zastoupeny vzorové hodnoty rádců a poté hodnoty pro automatického obchodníka. *Specimen* reprezentuje vzor pro daný typ obchodní strategie.

7.5.2 Účelová funkce

Účelová funkce definuje uspořádání nad množinou možných řešení, v našem případě potřebujeme upřednostnit výsledky simulace strategie stejného typu[31], které jsou z našeho pohledu lepší. V této práci, a v simulačním programu, byla zvolena hodnota konečného zisku rovna hodnotě účelové funkce, což jednoduše znamená, že strategie, která vydělá více peněz než jiná strategie, bude upřednostněna při výběru v optimalizačních algoritmech[32].

7.6 Verifikace vlastností simulačního modelu

Verifikovat simulaci není vždy úplně triviální činností. V tomto případě je dobré porovnat požadované vlastnosti systému se simulačním modelem a kriticky zhodnotit dosažený stav. Přehled vlastností simulačního modelu je znázorněn v tabulce 5.

[30] Knihovna je dostupná pro adresou: `https://github.com/vaclavnemec/jExtreme`

[31] Například strategie využívající indikátory SMA a RSI k obchodování s akciemi společnosti Google

[32] Možným vylepšením by mohlo být kalkulování s maximálním ztrátovým obchodem, což by snížilo možné riziko při obchodování

Vlastnosti simulace		
F	Simuluje obchodování na burze za použití historických dat	Ano
Q	Simulace je architektonicky rozdělena, aby bylo možné systém jednoduše rozšířit a upravovat	Ano
F	Při obchodování bude možné nastavit stop loss	Ano
F	Při obchodování bude možné nastavit stop loss	Ano
F	Každý uzavřený obchod bude zpoplatněn pomocí poplatku	Ano
F	Hodnota poplatku je nastavitelná	Ano
F	Simulace umožňuje sestavení obchodní strategie kombinací jednotlivých strategií	Ano
F	Je možné optimalizovat strategie pomocí optimalizačních algoritmů	Ano
F	Simulace umožňuje obchodování pomocí indikátorů	Ano
F	Simulace umožňuje obchodování pomocí neuronové sítě	Ano
F	Simulace umožňuje obchodování pomocí indexu *nálady trhu*	Ano
F	Je možné využít kombinaci indikátorů a jiných rádců	Ano
F	Každý typ rádce lze použít vícekrát	Ne. Nelze například použít indikátor SMA dvakrát s jiným nastavením. Vhodné pro rozšíření.
F	Posouvání *stop loss* hodnoty	Ne. Vhodné pro rozšíření.

Tabulka 5: Verifikace požadavků. Q znamená kvalitativní a F funkcionální.

Obrázek 15: Rozdělení uživatelského rozhraní - Hlavní panel

7.7 Uživatelské rozhraní

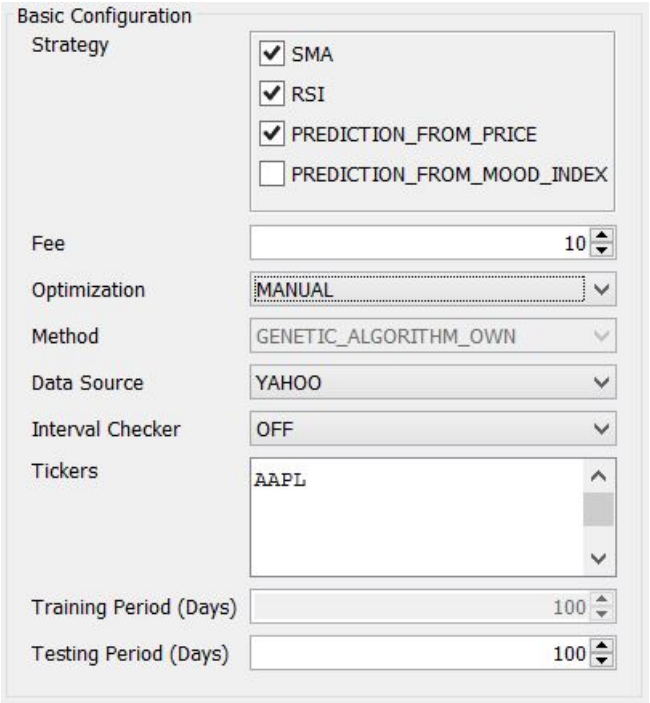
Uživatelské rozhraní aplikace je napsáno v desktopovém frameworku Swing, který je základní součástí JDK. Uživatelské rozhraní využívá simulační proces popsaný v předešlých podkapitolách přímo a přidává několik jednoduchých funkcí, jako je práce s grafy pomocí knihovny JFreeChart [33] a automatické ukládání výsledků pro pozdější analýzu pomocí serializace na disk. V uživatelském rozhraní je možné nakonfigurovat parametry simulace tak, jak byly popsány v kapitole 7.4. Je taky možné spouštět jednu či více simulací zároveň, sledovat simulace pomocí textového výstupu a progress baru, nahlížet na výsledky simulace, porovnávat výsledky tickerů (podle pohybu na účtu) a nahlížet na archivní výsledky.

7.7.1 Hlavní panel

Rozdělení hlavní obrazovky uživatelského rozhraní je vidět na obrázku 15. Uživatelské rozhraní je rozděleno na tři hlavní části. Těmi jsou vrchní pracovní lišta, levý konfigurační panel a pravý panel s kartami. Na pracovní liště jsou tlačítka “Run Simulation”, “Open Report” a “Mood Test”. Tlačítkem “Run Simulation” se spouští simulace nakonfigurovaná na levém panelu. Tlačítkem “Open Report” se otevírá dialogová karta pro otevření uloženého reportu. Tlačítkem “Mood Test” nastavíme na levý konfigurační panel hodnoty tickerů, které jsou v této práci dále použity pro otestování indexu *nálady trhu*.

7.7.1.1 Konfigurační panel Levý panel je dále rozdělen na čtyři menší panely, ze kterých jsou v jeden okamžik viditelné maximálně 3, podle toho, co jsme nastavili na panelu “Basic Con-

[33] `http://www.jfree.org/jfreechart/`



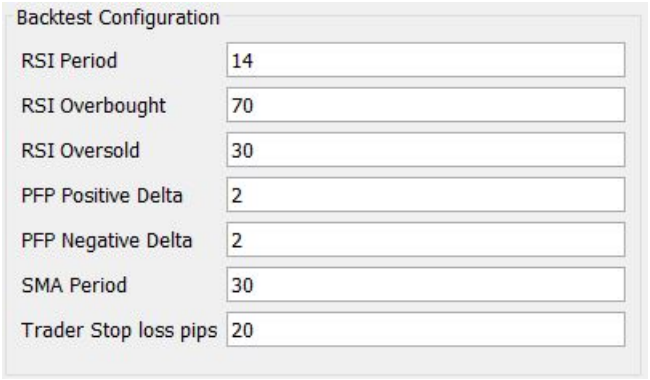
Obrázek 16: Panel "Basic Configuration"

figuration", který je nejvíce na vrchu a je viditelný vždy. Položka **Strategy** Obrázek 16 zobrazuje panel "Basic Configuration".

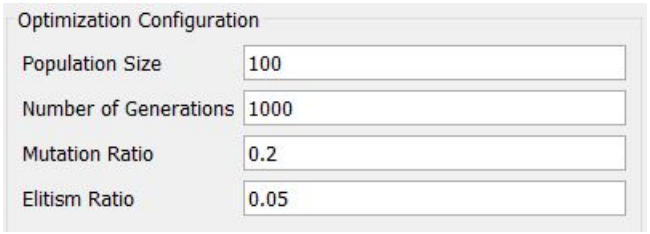
Nastavení je shodné s podmnožinou parametrů simulačního požadavku a jejich význam lze pochopit po přečtení kapitoly 7.4, u některých ovládacích prvků je ale nutné uvést způsob, jakým se s nimi pracuje. Část panelu, označená jako **Strategy**, slouží k výběru částí strategie, tzn. volí se tzn. rádcové automatického obchodníka. Strategii je možné poskládat z indikátorů SMA a RSI. Je také možné zvolit predikci z ceny, popřípadě predikci z indexu *nálady trhu*. Strategii lze libovolně nakombinovat. Výběr, zda-li chceme optimalizovat či ne, se provádí pomocí položky označené jako **Optimization**. V případě, že je vybráno ON, zviditelní se panel nastavení optimalizace. V opačném případě je viditelný panel pro manuální zadání parametrů strategie. Optimalizační metoda se vybírá v poli **Method** a když se hodnota v tomto polí změní, aktualizuje se *panel nastavení konfigurace* popsaný v podkapitole 7.7.1.3. Pokud je optimalizace zapnutá, viz předchozí položka Optimization, vybírá uživatel optimalizační metodu pomocí tohoto ovládacího prvku, viz *algoritmus* v 3. Pole **Tickers** slouží k zadání tickerů, které oddělujeme čárkou. Každý ticker musí vyhovovat regulárnímu výrazu `^[A-Z\\-]+$`.

7.7.1.2 Panel manuální konfigurace

Panel manuální konfigurace slouží ke zvolení parametrů strategie manuálně. Je viditelný pouze tehdy, pokud je vypnutá optimalizace. Na obrázku 17 jde vidět, jak panel vypadá. V levé části je textový popis položky a v pravé textové pole. Popis položky se zkládá ze zkratky *rádce* a jeho konkrétní vlastnosti. Všechny parametry jsou uvedeny v tabulkách 1 a 2. Položky/parametry, viditelné na tomto panelu, jsou právě ty parametry, které



Obrázek 17: Panel manuální konfigurace



Obrázek 18: Panel nastavení optimalizace

by byly optimalizovány, pokud by byla optimalizace zapnutá. Panel je viditelný v případě, že je optimalizace vypnutá.

7.7.1.3 Panel nastavení optimalizace

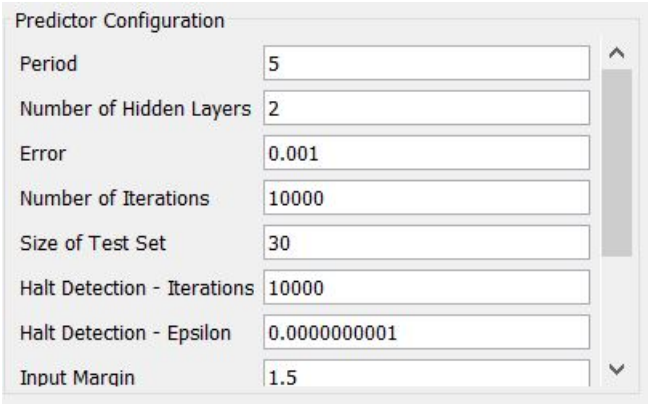
Panel nastavení optimalizace slouží ke konfiguraci vybrané optimalizační metody. Panel je znázorněn na obrázku 18. Všechny položky jsou uvedeny v tabulce 3. Položky v tomto panelu jsou automaticky aktualizovány po změně hodnoty **Method** v panelu základní konfigurace.

7.7.1.4 Panel nastavení predikce

Nastavení predikce je v našem případě shodné s nastavením neuronové sítě. Panel je znázorněn na obrázku 19 a hodnoty jsou popsány v tabulce 4. Panel je viditelný, pokud požadujeme, aby součástí strategie byla predikce z ceny.

7.7.2 Panel s kartami

Tento panel obsahuje dvě úrovně karet (tabů), oblast s grafy, tabulku provedených transakcí a *combo box* pro výběr tickeru. Na tomto panelu můžeme sledovat průběh simulace v kartě "Output #n", kde n je korelační identifikátor simulace. Karta obsahuje textové pole a *progres bar* ve spodní části. Důležitější kartou je druhý typ karty, a tím je "Report #n", kde n je analogicky také korelační identifikátor simulace. Karty se stejným identifikátorem patří stejné simulaci. Report obsahuje druhou úroveň karet s kartami "Backtest Chart", "Training Chart" a "Output". První dvě zmíněné karty jsou obsahově identické, liší se pouze data, nad kterými byla simulace provedena. "Backtest



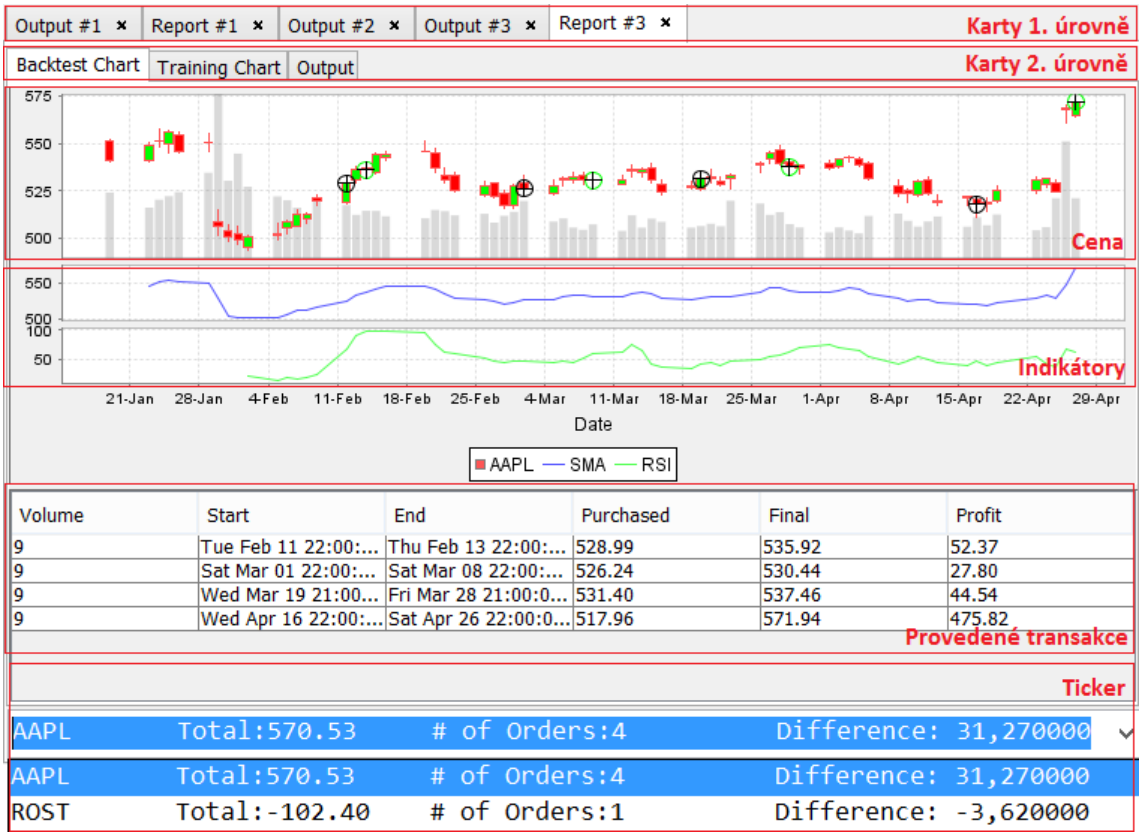
Obrázek 19: Panel “Predictor Configuration”

Chart” operuje s testovací množinou dat, to znamená, že simulace nezná dopředu data. Opačně pak “Training Chart” je stejná strategie aplikovaná na trénovací množinu; tato karta je zobrazena pouze v případě, že jsme prováděli optimalizaci (potřebovali jsme strategii trénovat). Obě karty obsahují zobrazení ceny ve svíčkovém grafu a zobrazení indikátorů v liniovém grafu, k tomuto účelu bylo využito knihovny jFreeChart http://www.jfree.org/jfreechart/. Pod oblastí s grafy je zobrazena tabulka provedených transakcí. Jde vidět, s jakým objemem bylo obchodováno, doba, po kterou byl objem vlastněn, nákupní a prodejní cena a výsledný profit. Posledním ovládacím prvkem na této kartě je *combo box*, kterým přepínáme jednotlivé odsimulované tickery. Při změně hodnoty na tomto ovládacím prvku se automaticky aktualizuje obsah celé karty. Hodnoty jsou šeřazeny podle získaného kapitálu. Karta “Output” obsahuje textový výpis, ze kterého je například zřejmé, jaké hodnoty parametrů byly nalezeny při optimalizaci. Pravý panel s kartami je zobrazen na obrázku 20

7.8 Spuštění programu

Pro spuštění simulačního programu je nutné mít nainstalovanou JDK[34], minimálně od verze *1.8.0* a buildovací nástroj Maven [35] ve verzi *3.0.4* a mít přístup k internetu, aby bylo možné stáhnout požadované závislosti a aby měla simulace přístup k historickým datům. Přiložený nosič obsahuje soubory *install.bat*, *start.bat* a *corr.bat*. První soubor slouží ke kompilaci zdrojových souborů, druhý soubor ke spuštění užibatelského rozhraní aplikace a poslední soubor slouží k exportu dat pro korelační analýzu.

[34] `http://www.oracle.com/technetwork/java/javase/downloads/index.html`
[35] `http://maven.apache.org/`



Obrázek 20: Panel s kartami

8 Simulace

V této kapitole jsou uvedeny výsledky, které byly odsimulovány pro 3 druhy strategií, srovnání strategií a na závěr výpočet *korelačního koeficientu* pro index *nálady trhu*. Prvním druhem strategie je klasická strategie využívající kombinaci indikátorů SMA a RSI. Druhým typem strategie je strategie využívající neuronovou síť pro predikci ceny a posledním typem strategie je strategie využívající vytvořený *index nálady*. Simulační framework nám umožňuje kombinovat všechny možnosti dohromady, taková strategie by ale nefungovala dobře, protože komponenta *automatický obchodník* vstupuje do pozice jen tehdy, pokud mu to radí všichni *rádci*, viz podkapitola 7.3.4.

8.1 SMA a RSI

Klasický přístup pro obchodování s využitím technické analýzy používá technických indikátorů. V této simulaci jsou využity dva indikátory a to SMA, což je indikátor sledující trend a RSI, což je vedoucí indikátor, viz kapitola 2. Tickery pro experiment byly vybrány namátkou, byla spuštěna simulace s optimalizací a její výsledky jsou vidět v příloze v tabulce 9. Parametry spuštění jsou v příloze v tabulce 8, množina odsimulovaných tickerů je patrná z výsledkové tabulky. Trénovací množina začíná datem 2. srpna 2011 a končí 28. dubna 2014 v den spuštění testu.

Po prozkoumání výsledků bylo zjištěno, že strategie jsou výdělečné, ale že většina hodnot akcií stoupala, takže by bylo pro obchodíka v průměru mnohem výhodnější pasivně držet akcie a vydělal by více, což je pravděpodobně dáno opatrností strategie. K povšimnutí stojí ticker IGT. Hodnota akcií klesala a za posledních 1 000 dní se propadla o 6.08 dolarů, avšak strategie generovala vysoký zisk. Proto byly pro druhý experiment vybrány takové akciové tituly, které v poslední době ztrácely, abychom potvrdili, že je strategie pro takové situace odolnější. Druhý pokus byl tedy zaměřen na tituly ADT, AVP, FE, IGT, NEM, RIG a LUK, přičemž ostatní parametry simulace zůstaly stejné. Výsledy potvrdily předchozí úvahu, že i v případě celkového poklesu ceny akcie dokáže simulace nalézt strategii, která je robustní a generuje zisk, popřípadě minimalizuje ztrátu, a to velmi úspěně (s výjimkou posledních dvou strategií). Výsledky jsou v tabulce 10.

8.2 Predikce ceny pomocí neuronové sítě

V této části simulace byla otestována strategie založená na predikci neuronové sítě. Parametry strategie jsou ukázány v tabulce 11. Oproti předchozí simulaci jsou obě množiny, jak testovací, tak trénovací, zkráceny na 100 dní. Důvodem je citlivost neuronové sítě. Trénovací množina se pro naučení neuronové sítě nepoužije celá, ale použije se pseudonáhodný výběr. Čím je trénovací množina vyšší, tím klesá pravděpodobnost, že se neuronová síť naučí, data začínají být velmi odlišná a při učení se nedocílí požadované maximální chyby. Akciové tituly pro tuto simulaci byly vybrány náhodně. Datum spuštění testu bylo 1. května 2014. Tabulka 12 zobrazuje výsledky simulace. Strategie využívající neuronovou síť generovala ve všech případech zisk, popřípadě mnohem menší ztrátu, než kdybychom pasivně drželi pozici. Je dobré si všimnout, že strategie generuje velmi málo signálů pro nákup a prodej. Trénovací množina začíná datem 16. ledna 2014 a končí 1. května 2014 v den testu.

Ticker	Dnů	Změna ceny	Index nálady		Neuronová síť		RSI a SMA	
			Zisk	Obchodů	Zisk	Obchodů	Zisk	Obchodů
MUR	91	5.99	0	0	515.48	1	328	2
WDC	43	2.21	118.18	1	5.66	1	0	0
WMB	31	1.59	0	0	301.19	1	0	0
EQR	65	0.65	45.25	1	106.45	1	0.65	1
L	59	0.82	0	0	0	0	0	0
LNC	84	0.42	0	0	-0.50	3	0	0
FLR	72	-4.43	-226.38	1	98.34	2	-136.48	1
RL	65	-8.62	-544.44	1	0	0	0	0
PCP	107	-17.72	-386.64	5	-24.0	1	-392	1

Tabulka 6: Srovnání strategií

8.3 Index nálady trhu

V této části simulace bylo cílem otestovat vytvořený index *nálady trhu* a jeho účinnost. Parametry spuštěné simulace jsou vidět v tabulce 13. V případě, že se využívá nálady ze zpráv, se parametry simulace *trainingDays* a *testingDays*, tzn. rozdělení na trénovací a testovací množinu, vypočítají dynamicky podle dostupnosti zpráv a poté se množina zpráv rozdělí symetricky na trénovací a testovací množinu. Datum spuštění testu bylo 1. května 2014.

Výsledek simulace pro vybrané tickery je vidět v tabulce 14. Z tabulky je patrné, že valná většina namátkově vybraných titulů v testovacím období posílila. Optimalizované strategie často jenom pasivně držely podíl a z toho důvodu jsou hodnoty ve sloupích *Zisk* a *Při pasivním držení* stejné. Výjimkou jsou tituly JCP, KR, VNO, CTXS, MDT, PH, JNJ, XOM, DTV, TXT a UPS, pro které se strategie chovala jinak a dosáhly jiného výsledku, než při pasivním držení, z čehož prvních 6 titulů bylo obchodováno s vyšším ziskem než při pasivním držení a posledních 5 titulů se ztrátou.

Následně byly vybrány tituly, které za poslední dobu spíše oslabily. Výsledek je vidět v tabulce 15. Výsledky obchodování se strategií využívající indexu *nálady trhu* ukázaly, že index není použitelný, pokud bude strategie využívat pouze hraniční hodnoty pro určení pozitivních či negativních zpráv. Možným vylepšením by bylo upravení strategie tak, že by bylo posledních *n* hodnot indexu přivedeno na vstup naučené neuronové sítě, což není ve stávající implementaci možné bez úprav. Je taky možné, že zdroj finančních zpráv není dostatečně relevantní, a proto strategie nefunguje a jeho nahrazením za jiný zdroj dat, například sociální síť Twitter [36], bychom dosáhli mnohem lepších výsledků.

8.4 Srovnání strategií

V této části byly srovnány všechny simulace na stejných testovacích datech. Nejdříve bylo nutné odsimulovat strategii založenou na indexu *nálady trhu*, protože bylo nutné zjistit interval testovací množiny, který je závislý na dostupnosti zpráv. Poté byly postupně spuštěny další simulace pro stejnou testovací množinu. Výsledky jsou vidět v tabulce 6. Testy byly spuštěny 1. května 2014.

[36] `https://twitter.com/`

Ticker	Koeficient	Ticker	Koeficient
AEP	-0.2213853	KR	0.08140277
AMGN	-0.1073615	MDT	-0.005033599
CTXS	0.08259012	PPG	-0.278307
CVX	-0.5217822	SO	-0.07408605
DTV	0.4928555	TXT	0.1454422
EQT	-0.04060295	UPS	0.2809409
GE	-0.4728908	VNO	0.03588778
JCP	0.3992919	WM	0.0743098
JNJ	0.190687	XOM	-0.8916955

Tabulka 7: Korelační koeficient změny denní ceny a nálady akciového titulu

Z tabulky 6 je patrné, že nejlépe obchodovala strategie využívající *neuronovou síť*. Častým jevem je pasivita strategií, v mnoha případech neproběhl ani jeden obchod, to je pravděpodobně dáno zkrácením testovací periody na dobu dostupnosti historických zpráv. Hlavně strategie *RSI a SMA* může být ovlivněna velmi výrazně, protože *většinou* využívá delší periodu a trvá déle, než začně generovat signály k nákupu či prodeji. Informace o počtu dní je důležitá, protože první strategie je závislá na dostupnosti zpráv, a tím definuje délku trénovací množiny pro ostatní strategie, aby bylo srovnání směrodatné.

8.5 Korelační koeficient nálady trhu a ceny

Posledním experimentem, provedeným v rámci této práce, bylo vypočtení korelačního koeficientu pro hodnoty indexu *nálady trhu* a změny ceny akcie. K tomuto účelu byla data vyexportována do formátu *csv*. Výpočet korelačního koeficientu byl proveden v prostředí *R*[37]. Kód, který umožní hromadný výpočet korelačního koeficientu, je vidět ve výpisu 7, údaje o každém tickeru byly uloženy v samostatném souboru, například AEP.csv.

```
korelacni_koeficient <- function(filename) {
  values = read.csv(filename, header=FALSE)

  print (filename)
  print (cor.test (values [,1], values [,2]) )

  setEPS()
  postscript (paste(gsub(".csv", "", filename), "eps", sep = ".") )
  plot (values [,1], values [,2], main = filename, xlab="Cena", ylab="Nalada")
  dev.off ()
}
mapply(korelacni_koeficient, list . files (pattern = "^[ A-Z]+.csv$"))
```

Výpis 7: Výpočet korelačního koeficientu hromadně pro více titulů s exportem grafů v R

[37] http://www.r-project.org/

Z výsledků v tabulce 7 a z grafů v příloze je patrné, že neexistuje silná korelace u žádného z testovaných tickerů. U tickeru XOM se zdá, že je korelace vysoká podle vysoké hodnoty koeficientu, ale z grafu vyplývá, že je výpočet koeficientu založen na třech hodnotách, což je nedostačující.

9 Závěr

9.1 Zhodnocení výsledků

Hlavním cílem této práce bylo vytvoření systému pro simulaci burzovního obchodování, v rámci které bude možné vytvořit a optimalizovat obchodní strategie pomocí evolučně inspirovaných algoritmů, vytvoření obchodního *indexu nálady* za použití finančních zpráv a vyhodnocení a srovnání úspěšnosti nalezených strategií.

Proto, abych důkladně pochopil danou problematiku, jsem musel nejdříve provést studii, která se týkala technické analýzy, indikátorů používaných v technické analýze a způsobu jejich využití. Touto studií se zabývá kapitola 2. Dále byla v kapitole 3 provedena rešerže systémů, které nabízejí možnost automatického obchodování a bylo zmíněno, co přesně automatické obchodování znamená. Dalším tématem, které bylo nutné pochopit, byla oblast matematické optimalizace. V rámci této práce byla vyvinuta knihovna jExtreme[38], která byla vydána jako Open Source pod licencí GNU Version 3. Optimalizací se zabývá kapitola 4. Obchodní strategie může být založená na predikci časové řady, a proto bylo nutné nastudovat problematiku neuronových sítí, které jsou popsány v kapitole 5. Pro práci s internetovými zprávami bylo nutné nastudování oblasti týkající se zpracování přirozené řeči, tato problematika je nastíněna v katitole 6. V poslední řadě byl v rámci kapitoly 7 navržen a implementován simulační systém umožňující tvorbu obchodní strategie využívající obchodní indikátory, predikci ceny pomocí neuronové sítě a navržený index *nálady trhu* využívající zpracování přirozené řeči, konkrétně internetových finančních zpráv a její následnou optimalizaci a testování.

Z výsledků simulací a experimentů v kapitole 8 je jasné, že klasický přístup, založený na práci s indikátory, je lepší, než přístup, založený na práci s indexem *nálady trhu* v takové formě, v jaké byl implementován v simulačním programu. Byly vytipovány možné příčiny selhání, kterými je pravděpodobně jednoduchost strategie pracující s hraničními hodnotami pro určení pozitivních a negativních zpráv, popřípadě kvalita zdroje dat, jímž byl server Finviz [39]. Možným vysvětlením pro selhání navrženého indexu *nálady trhu* je fakt, že zkoumaná nálada zpráv odráží vývoj cen akciových titulů a nelze provádět predikci opačným směrem.

9.2 Další vývoj

Touto prací byl položen základní kámen pro vytvoření celé obchodní platformy, která by mohla sloužit pro automatické obchodování, tvorbu strategií a objevování nových způsobů v obchodování. Možným rozšířením je například používání rozdílných strategií pro vstup a výstup z, respektive do, pozic, popřípadě implementování principů *money managementu*, nebo techniky, při které automatický obchodník automaticky posouvá hodnotu *stop loss*, což může vést k vyšším ziskům. V této práci byl implementován index *nálady trhu*, ten by však mohl být dále využit i v jiných oblastech, ne jenom ve sféře burzovní obchodování. Obecně je totiž možné extrahovat náladu jakékoliv entity z jakéhokoliv typu textů.

Bc. Václav Němec

[38] `https://github.com/vaclavnemec/jExtreme`
[39] `www.finviz.com`

10 Reference

[1] Volatility Definition. In: Investopedia [online]. [cit. 2014-04-28]. Dostupné z: http://www.investopedia.com/terms/v/volatility.asp

[2] Support a resistance. In: Finančník [online]. 2004 [cit. 2014-04-28]. Dostupné z: http://www.financnik.cz/komodity/manual/komodity-support-resistance.html

[3] Options Basics: What Are Options?. In: Investopedia [online]. [cit. 2014-04-28]. Dostupné z: http://www.investopedia.com/university/options/option.asp

[4] ACHELIS, Steven B. Technical analysis from A to Z. 1nd edition. 2014. ISBN 978-007-1826-297.

[5] Klouzavé průměry. In: www.financnik.cz [online]. 2009 [cit. 2014-02-20]. Dostupné z: http://www.financnik.cz/komodity/zkusenosti/forex-klouzave-prumery.html

[6] Relative Strength Index. In: Http://stockcharts.com/ [online]. [cit. 2014-02-21]. Dostupné z: http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:relative_strength_index_rsi

[7] Algo Trader [online]. [cit. 2014-02-22]. Dostupné z: https://code.google.com/p/algo-trader/

[8] Evoluční výpočetní techniky: principy a aplikace. 1. české vyd. Praha: BEN, 2009, 534 s. ISBN 978-80-7300-218-3.

[9] Zachyťte ten správný moment: indikátor Momentum. Www.financnik.cz [online]. 2005 [cit. 2014-02-21]. Dostupné z: http://www.financnik.cz/komodity/findikatory/indikator-momentum.html

[10] Introduction to Technical Indicators and Oscillators. StockCharts.com - ChartSchool [online]. [cit. 2014-02-22]. Dostupné z: http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:introduction_to_tech

[11] Forex robot (AOS): Automatický obchodní systém. Svět obchodování na FOREXu [online]. [cit. 2014-02-22]. Dostupné z: http://www.fxstreet.cz/forex-robot-aos-automaticky-obchodni-system.html

[12] Cyan Spring [online]. [cit. 2014-02-22]. Dostupné z: http://www.cyanspring.com/

[13] VONDRÁK, Ivo. Umělá inteligence a neuronové sítě. Ostrava: VŠB, 1995, 139 s. ISBN 80-707-8259-5.

[14] Neural Networks for Time Series Prediction. In: Artificial Neural Networks [online]. 2006 [cit. 2014-02-28]. Dostupné z: http://www.cs.cmu.edu/afs/cs/academic/class/15782-f06/slides/timeseries.pdf

[15] Natural Language Processing. In: LILLY, Elizabeth D. Syracuse University Library [online]. 2001 [cit. 2014-02-24]. Dostupné z: http://surface.syr.edu/cgi/viewcontent.cgi?article=1043&context=istpub

[16] Introduction to Sentiment Analysis. In: LCT [online]. [cit. 2014-03-02]. Dostupné z: http://www.lct-master.org/files/MullenSentimentCourseSlides.pdf

[17] Arizona Financial Text System. Dr. Robert P. Schumaker [online]. [cit. 2014-03-02]. Dostupné z: http://robschumaker.com/research/azfintext/

[18] SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In: [online]. [cit. 2014-04-30]. Dostupné z: http://nmis.isti.cnr.it/sebastiani/Publications/LREC06.pdf

[19] Reviews Classification Using SentiWordNet Lexicon [online]. [cit. 2014-03-02]. Dostupné z: http://www.academia.edu/1336655/Reviews_Classification_Using_SentiWordNet_Lexicon

[20] Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In Proceedings of HLT-NAACL 2003, pp. 252-259.

[21] MARCUS, Mitchell a Mary Ann MARCINKIEWICZ. Building a Large Annotated Corpus of English: The Penn Treebank. In: [online]. 1993 [cit. 2014-04-30]. Dostupné z: http://acl.ldc.upenn.edu/J/J93/J93-2004.pdf

A Parametry simulací a výsledky

Parametr	Hodnota
Strategie	SMA a RSI
Poplatek	10
Optimalizační metoda	Genetický algoritmus
Velikost populace	100
Počet generací	1 000
Pravděpodobnost mutace	0.2
Poměr elitních jedinců	0.05
Zdroj dat	YAHOO
Kontrola intervalu	Vypnuto
Délka trénovací množiny	1 000 dní
Délka testovací množiny	1 000 dní

Tabulka 8: Parametry simulace SMA a RSI

Ticker	Zisk	Počet obchodů	Změna ceny	Při pasivním držení
BA	4112.61	4	58.33	4141,43
EOG	3298.66	3	-2.58	-126,42
PFG	3173.32	8	18.87	3434,34
UNH	2493.28	14	27.64	2874,56
HD	2406.04	17	45.13	6588,98
TXT	2098.50	2	16.99	3822,75
ECL	1846.00	5	55.46	5546,0
AEP	1718.20	6	15.53	2049,96
BRK-B	1550.93	6	52.33	3506,11
USB	1464.08	4	14.24	2734,08
EMR	1256.00	4	19.38	1976,76
CF	1214.00	7	83.44	2670,08
GE	1038.68	5	8.63	2399,14
MDT	912.51	9	23.33	3336,19
IGT	859.80	8	-6.08	-1641,6
AAPL	834.91	3	175.19	2277,47
GILD	741.60	2	32.38	3885,6
MO	677.36	14	12.96	2475,36
UPS	615.26	3	29.78	2203,72
ED	550.05	10	4.99	474,05
PEP	524.36	4	21.39	1668,42
PPG	487.80	5	109.08	6544,8
CCL	370.56	24	5.25	798,
DHR	294.62	4	24.62	2535,86
VNO	56.38	3	8.45	456,3
XOM	31.59	5	20.81	1311,03
BAC	-96.70	1	6.14	3131,4
NKE	-459.04	5	-16.64	-931,84
KR	-880.20	10	20.46	4153,38
JCP	-1151.44	1	-22.47	-3685,08
Trénovací množina začíná datem 2. srpna 2011 a končí 28. dubna 2014				

Tabulka 9: Náhodně vybrané akciové tituly obchodované kombinací indikátorů SMA a RSI.

Ticker	Zisk	Počet obchodů	Změna ceny	Při pasivním držení
IGT	867.80	13	-6.08	-1641.6
LUK	781.19	1	-7.87	-1172.63
AVP	-111.76	3	-11.01	-2124.93
RIG	-222.28	4	-18.59	-1505.79
FE	-1100.72	2	-10.18	-1140.16
NEM	-2811.30	26	-28.88	-2599.2
Trénovací množina začíná datem 2. srpna 2011 a končí 28. dubna 2014				

Tabulka 10: Výsledky obchodování strategie SMA a RSI na ztrátových titulech.

Parametr	Hodnota
Zdroj dat	Yahoo
Poplatek	10
Kontrola intervalu	Vypnuto
Délka trénovací množiny	100 dní
Délka testovací množiny	100 dní
Perioda neuronové sítě	5
Počet skrytých vrstev sítě	2
Chyba neuronové sítě	0.1
Maximální počet iterací při učení	10000
Počet migrací SOMA	400
Počet Velikost populace SOMA	20
Prt	0.1
Step	0.011
Path Length	3

Tabulka 11: Parametry simulace využívající predikci neuronové sítě

Ticker	Zisk	Počet obchodů	Změna ceny	Při pasivním držení
XOM	620.36	1	2.91	148.410
AAPL	578.96	1	42.41	381.690
EOG	477.00	2	-72.11	-2163.30
PH	413.54	1	-0.73	-28.470
TXT	345.84	1	2.88	40.320
GE	238.90	1	0.48	91.20
PFG	223.20	1	-0.83	-87.980
ECL	218.48	1	0.71	34.080
GILD	207.77	1	-2.56	-156.160
BA	57.55	1	-13.21	-462.350
HD	38.98	1	-1.13	-70.060
CTXS	31.31	1	-2.25	-182.250
AEP	0.00	0	6.53	0.0
UNH	0.00	0	1.79	0.0
AMGN	-12.36	2	-6.49	-272.580
IGT	-603.40	1	-5.48	-1512.480
Trénovací množina začíná datem 16. ledna 2014 a končí 1. května 2014				

Tabulka 12: Výsledky strategie využívající predikci neuronové sítě.

Parametr	Hodnota
Strategie	Index *nálady trhu*
Poplatek	10
Optimalizační metoda	Genetický algoritmus
Velikost populace	100
Počet generací	1 000
Pravděpodobnost mutace	0.2
Poměr elitních jedinců	0.05
Zdroj dat	YAHOO
Kontrola intervalu	Vypnuto
Délka trénovací množiny	Podle dostupnosti zpráv
Délka testovací množiny	Podle dostupnosti zpráv

Tabulka 13: Parametry simulace pro index *nálady trhu*

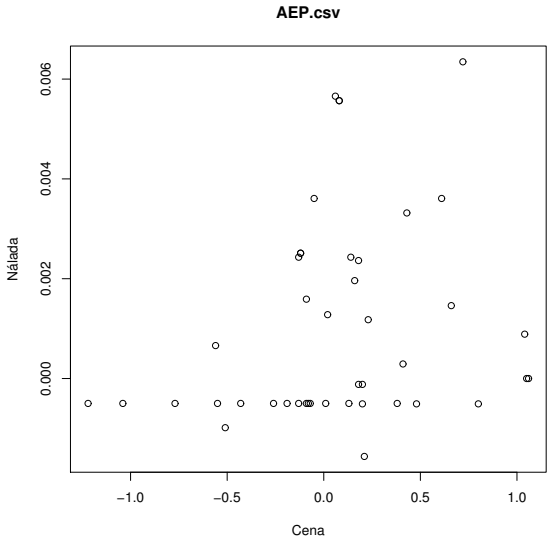
Ticker	Zisk	Počet obchodů	Změna ceny	Při pasivním držení
JCP	1174,00	1	0,18	211,320
IFF	1068,36	1	9,21	1068,360
KR	883,20	1	1,89	869,40
WM	671,00	1	2,75	671,0
EQT	594,36	1	11,43	594,360
VNO	511,02	6	2,99	304,980
CVX	476,80	1	1,49	476,80
SO	393,24	1	3,39	393,240
AEP	374,22	1	3,78	374,220
JNJ	279,00	3	1,34	335,0
MDT	182,32	3	0,65	167,70
CTXS	45,10	1	-2,36	-193,520
BRK-B	42,90	1	0,55	42,90
GE	0,00	1	0,0	0,0
XOM	-4,50	2	0,67	33,50
PH	-13,65	2	-4,09	-319,020
PPG	-147,50	1	-5,9	-147,50
DTV	-228,80	2	-0,03	-1,950
TXT	-238,00	7	2,42	987,360
CF	-840,94	1	-22,13	-840,940
UPS	-1748,00	2	-0,98	-931,0
AMGN	-2382,20	1	-5,54	-2382,20
				Simulace proběhla 1. května 2014

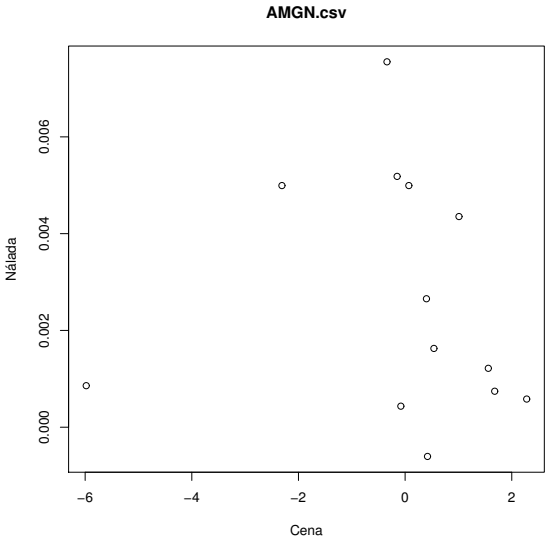
Tabulka 14: Obchodování pomocí indexu *nálady trhu*.

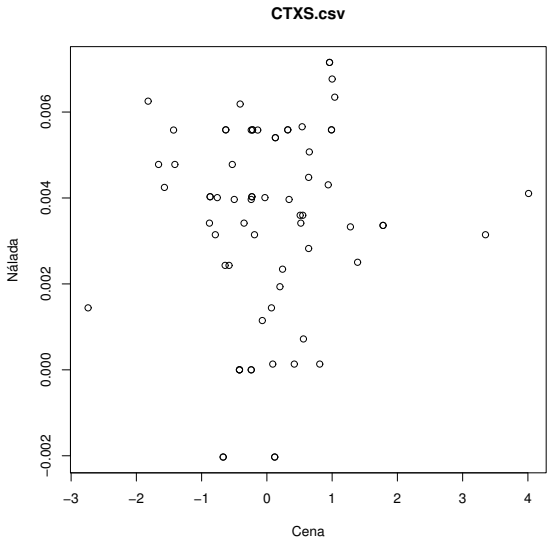
Ticker	Zisk	Počet obchodů	Změna ceny	Při pasivním držení
OI	585.32	1	1.21	595,32
TJX	-293.86	1	-1.71	-283,86
MA	-431.38	2	-3.07	-411,38
RTN	-600.94	1	-6.03	-590,94
NKE	-1374.76	1	-2.23	-1364,76
TYC	-2285.92	1	-2.18	-2275,92
				Simulace proběhla 1. května 2014

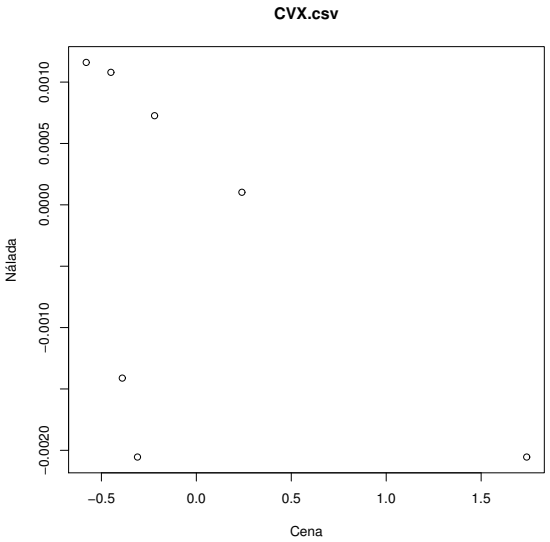
Tabulka 15: Obchodování pomocí indexu *nálady trhu* na titulech, které převážně oslabily.

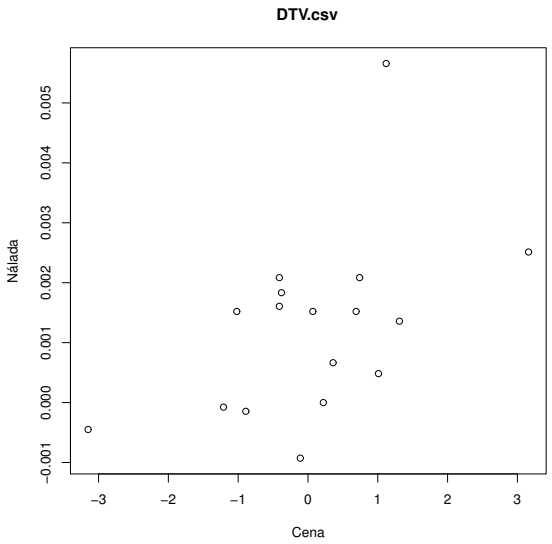
B Grafy korelace

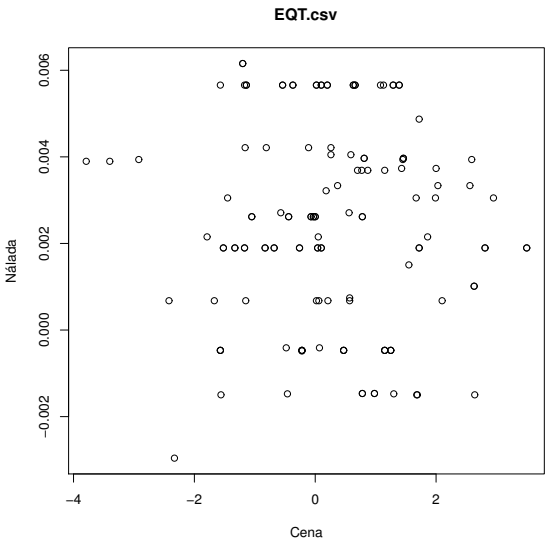


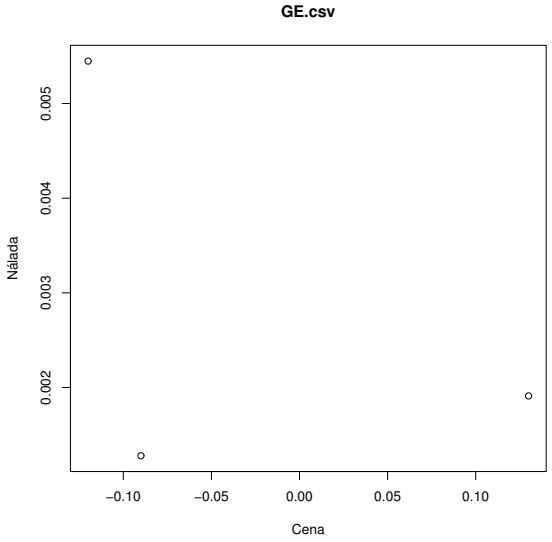


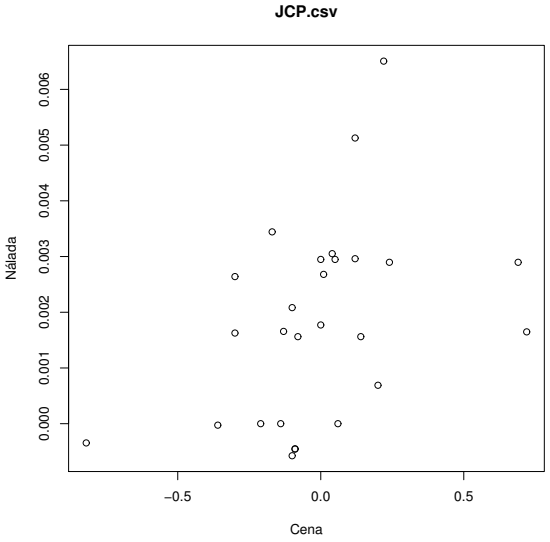


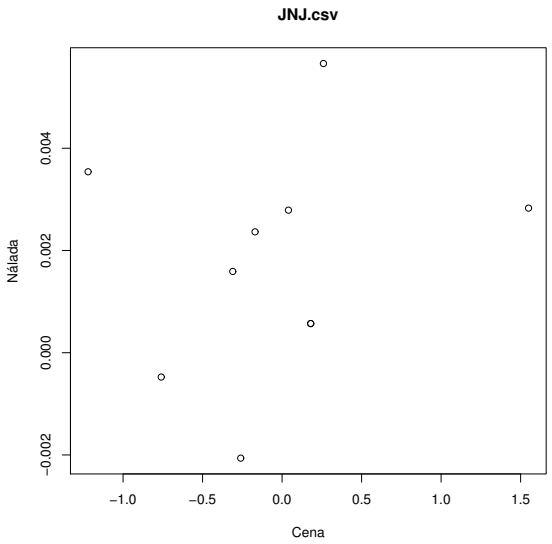


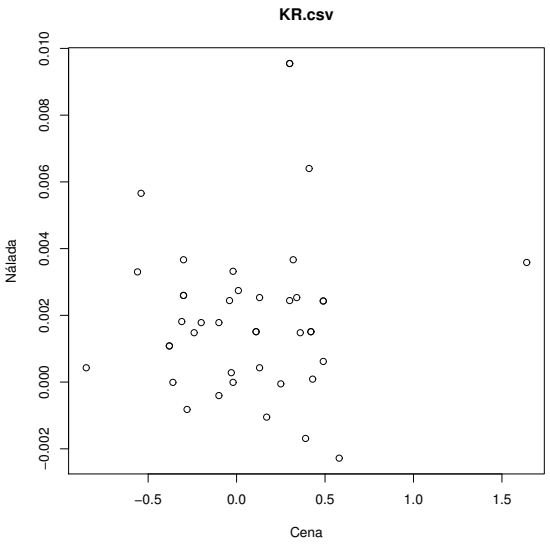

DTV.csv
Nálada
Cena
−0.001
0.000
0.001
0.002
0.003
0.004
0.005
−3
−2
−1
0
1
2
3

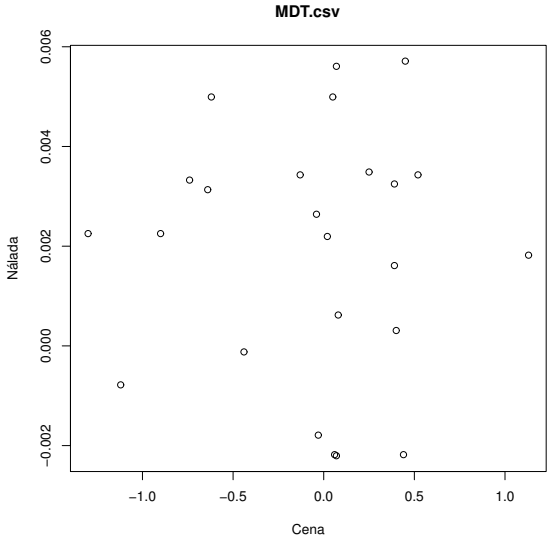

EQT.csv
Nálada
Cena
−0.002
0.000
0.002
0.004
0.006
−4
−2
0
2

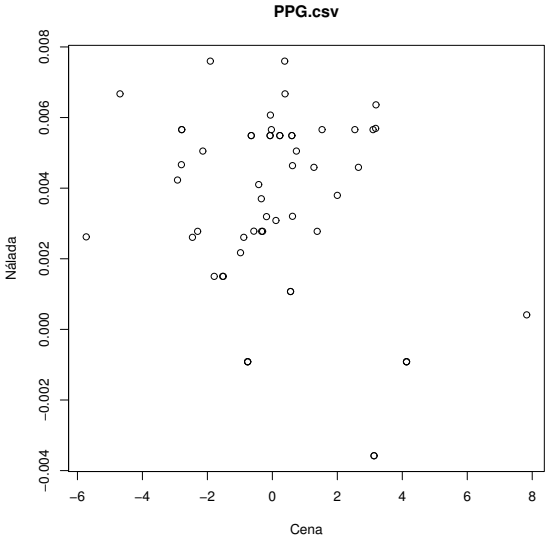

GE.csv
Nálada
Cena
0.002
0.003
0.004
0.005
−0.10
−0.05
0.00
0.05
0.10

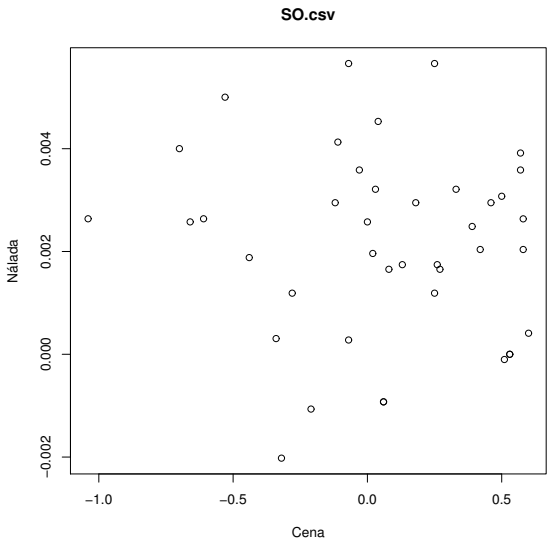

JCP.csv
Nálada
Cena
0.000
0.001
0.002
0.003
0.004
0.005
0.006
−0.5
0.0
0.5

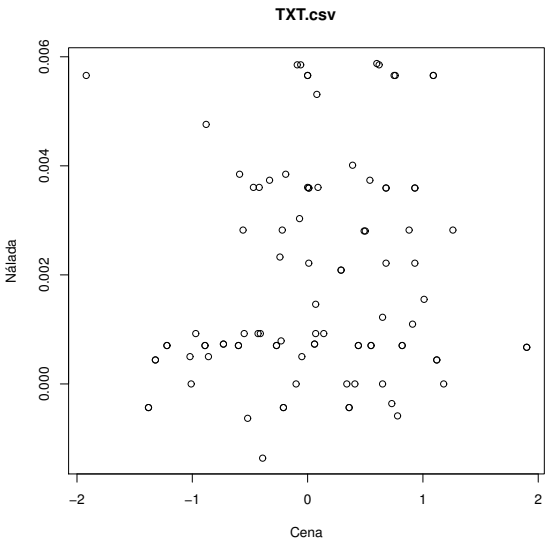

JNJ.csv
Nálada
Cena
0.004
0.002
0.000
−0.002
−1.0
−0.5
0.0
0.5
1.0
1.5

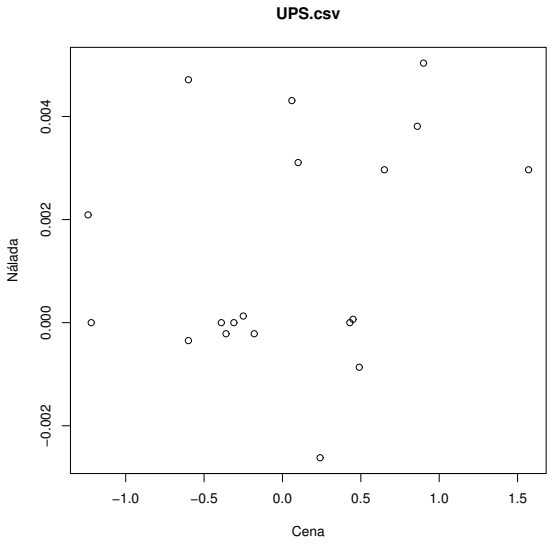

KR.csv
Nálada
Cena
0.010
0.008
0.006
0.004
0.002
0.000
−0.002
−0.5
0.0
0.5
1.0
1.5

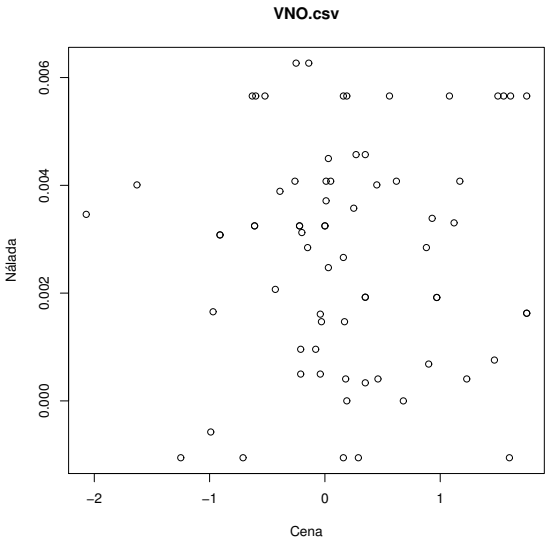

MDT.csv
Nálada
Cena
0.006
0.004
0.002
0.000
−0.002
−1.0
−0.5
0.0
0.5
1.0

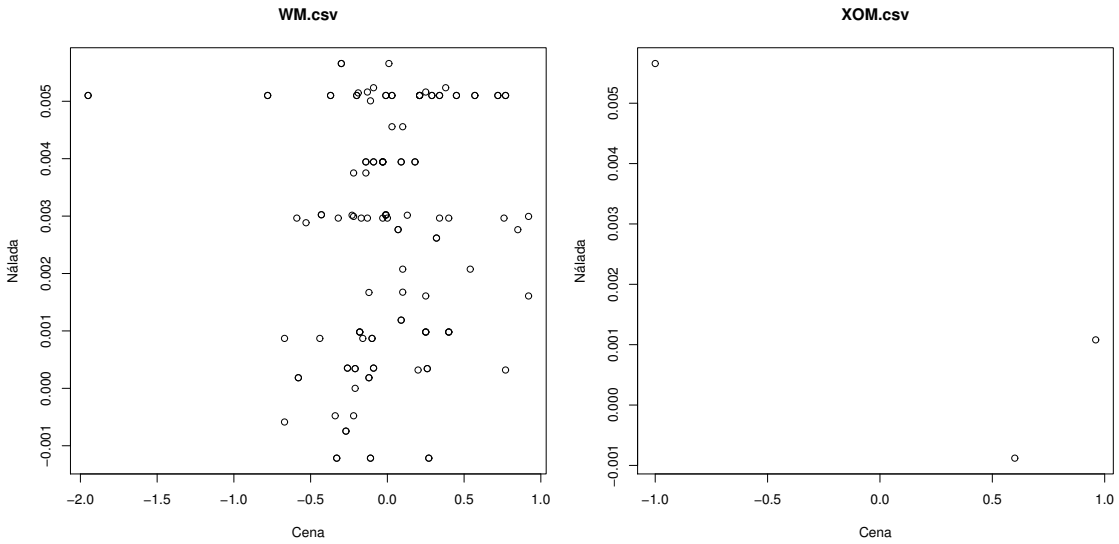

PPG.csv
Nálada
Cena
0.008
0.006
0.004
0.002
0.000
−0.002
−0.004
−6
−4
−2
0
2
4
6
8


SO.csv
Nálada
Cena
−0.002
0.000
0.002
0.004
−1.0
−0.5
0.0
0.5


TXT.csv
Nálada
Cena
0.000
0.002
0.004
0.006
−2
−1
0
1
2


UPS.csv
Nálada
Cena
−0.002
0.000
0.002
0.004
−1.0
−0.5
0.0
0.5
1.0
1.5


VNO.csv
Nálada
Cena
0.000
0.002
0.004
0.006
−2
−1
0
1


WM.csv
Nálada
Cena
−0.001
0.000
0.001
0.002
0.003
0.004
0.005
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
XOM.csv
Nálada
Cena
−0.001
0.000
0.001
0.002
0.003
0.004
0.005
−1.0
−0.5
0.0
0.5
1.0

C Obsah přiloženého CD

Přiložený nosič obsahuje následující adresářovou strukturu: